Universidad Carlos III de Madrid

DISEÑO E IMPLEMENTACIÓN DE UN PROXY PARA LA ELIMINACIÓN DE CONTENIDO ESTEGANOGRÁFICO

Proyecto de fin de Carrera

Tutor: Jorge Blasco Alís

Carlos Fernández Escolar

Ingeniería Técnica en Informática de Gestión

Agradecimientos

Quiero expresar mi más sincero agradecimiento a mi tutor, Jorge Blasco Alís, ya que gracias a todo el tiempo que ha dedicado a ayudarme, he podido realizar este proyecto de fin de carrera.

Quiero agradecer también a Celia el que siempre este ahí para apoyarme, darme energía y animarme.

Finalmente agradecer a mi madre y a mi familia el apoyo que me han dado, y el esfuerzo que han realizado durante mis años de estudio.

Índice de Contenidos

Índice de Figuras

Índice de Tablas

Capítulo 1
Introducción

Este capítulo sirve para ofrecer una introducción al proyecto. Se empezará explicando las motivaciones que llevaron a la realización de este proyecto, así como los objetivos que se desean alcanzar mediante la realización del mismo. Además, también se detallará la estructura del presente documento, con la intención de facilitar su lectura, explicando las partes de las que se compone. Para finalizar, el capítulo contiene un apartado dedicado al glosario de términos, compuesto por definiciones de las palabras claves del proyecto así como de los acrónimos que puedan aparecer en el mismo.

1.1 Motivación y Objetivos

La seguridad de los sistemas informáticos es muy importante para cualquier organización, bien sea por que es una herramienta importante en el área de negocios de dicha organización, o bien porque el sistema almacene información valiosa de uso privado. Dicha información puede verse comprometida debido a una amenaza externa, originada desde fuera de la red de la empresa, o por una amenaza interna, provocada desde dentro de la propia red.

La motivación para la realización de este proyecto es la eliminación de información oculta (canales ocultos), a través del protocolo HTTP, y que haya sido ocultada en algún archivo mediante el empleo de esteganografía. Esto puede servir para evitar la filtración de información desde dentro de la organización, o la entrada de datos no deseados a la red de ésta.

El proyecto tiene **dos objetivos principales, la creación de un proxy HTTP, y la implementación de filtros esteganográficos** para datos. El **proxy HTTP** es un intermediario entre un cliente y un servidor Web encargado de interceptar las transmisiones de red entre ambos, en este caso concreto, por motivos de seguridad. El proxy se encargara de interceptar todas las solicitudes y respuestas HTTP que se transmitan entre el cliente y el servidor, almacenando éstas para luego enviarlas a su destino inicial, previo filtrado de los datos enviados en las mismas. Tanto para el cliente como para el servidor Web la comunicación HTTP será una comunicación normal, viéndose únicamente afectada por un pequeño retardo producido por el tratamiento de la información, algo que no se produce en una comunicación sin proxy.

El segundo objetivo principal, la implementación de **filtros esteganográficos** para diferentes tipos de datos, consiste en la creación de una serie de filtros para la información enviada e interceptada por el proxy. Los filtros se encargarán de eliminar toda la información embebida mediante el uso de programas esteganográficos, eliminando dicha información mediante técnicas básicas para todos los tipos de datos como puede ser la técnica LSB, así como otras técnicas distintas implementadas por otros programas esteganográficos de libre distribución, y fáciles de encontrar en Internet. Dado que los filtros esteganográficos producen modificaciones en los datos filtrados, y ya que no se desea que por dichos filtrados se produzcan cambios significativos desde el punto de vista humano (cambios significativos en los colores de la imagen, en la calidad del audio, etc.), los filtros deberán respetar los datos originales evitando provocar distorsiones muy acentuadas en los mismos.

1.2 Estructura del Documento

Este apartado está dedicado a hacer una breve descripción de los capítulos que componen el documento, con el objetivo de tener una visión global del mismo. A continuación se puede ver el listado de capítulos que conforman el documento, junto a una descripción que detalla los temas tratados en éstos:

- Capítulo 1, **"Introducción"**: Este capítulo está formado por un resumen del proyecto, la estructura del documento del proyecto, así como por las definiciones y acrónimos de éste.

- Capítulo 2, **"Análisis del Proyecto":** Incluye una introducción al marco tecnológico en el que se encuentra el proyecto, así y como los requisitos de usuario y las pruebas de validación de éstos.

- Capítulo 3, **"Diseño del Proyecto":** Se compone de una descripción de la arquitectura del proyecto y las herramientas de desarrollo software usadas en el mismo. Además, incluye el modelado de la arquitectura, tanto estática como dinámica.

- Capítulo 4, **"Implementación del Proyecto":** Formado por los resúmenes detallados de la implementación de las distintas partes del proyecto. También contiene los resultados de las pruebas de validación realizadas al mismo.

- Capítulo 5, **"Gestión del Proyecto":** Contiene tanto la planificación como el presupuesto del proyecto.

- Capítulo 6, "**Conclusiones y Trabajos Futuros"**: Incluye las conclusiones del proyecto así como las futuras líneas de trabajo sobre el mismo.

- **"Referencias":** En este apartado se enumeran las referencias usadas en el proyecto.

- **"Anexos":** el documento cuenta con tres anexos. El primero de ellos es el manual de usuario, donde se detalla el método para instalar la aplicación y las utilidades de la misma. Los anexos restantes detallan los resultados de las pruebas realizadas a la aplicación.

1.3 Introducción a la Esteganografía

La esteganografía es la disciplina en la que se aplican técnicas para permitir ocultar mensajes u objetos dentro de otros, llamados portadores, con el fin de que la existencia del mensaje pase desapercibida. El origen de la palabra esteganografía deriva de la composición de dos palabras griegas, *steganos*, que significa cubierto u oculto, y *graphos*, que significa escritura. La esteganografía puede ser complementada con la criptografía con el fin de dar mayor seguridad a la información, cifrando previamente el mensaje que se quiere ocultar.

Distintas técnicas esteganográficas han sido usados a lo largo de los siglos. Herótodo en su libro, *Las Historias*, cuenta como enviaban mensajes tatuados en el cuero cabelludo de los esclavos, rasurándoles la cabeza, esperando a que les volviese a crecer el pelo para enviarlos con órdenes de rasurarles el pelo al llegar al destino. En fechas más recientes, durante la segunda guerra mundial, se usaron los microfilmes, ocultando mensajes en código Morse en los signos de puntuación o en los puntos de las letras i's. También a lo largo de la historia, y en la actualidad, se han usado tintas invisibles que reaccionaban al calor, a reacciones químicas o luz de determinada longitud de onda para la ocultación de mensajes.

En la esteganografía moderna, y al contrario que la esteganografía clásica, basada en desconocer el canal encubierto que se está usando, se usan canales digitales para la transmisión de los mensajes, tales como archivos de texto, audio, video e imágenes digitales, archivos ejecutables, etc. Actualmente existen una gran variedad de métodos para ocultar información dentro de archivos digitales. A continuación se muestran algunos de los más usados:

- **Enmascaramiento y filtrado (Watermarking):** La técnica de las "Marcas de agua" consiste en introducir información imperceptible en datos audiovisuales. Es un método todavía en crecimiento. A partir de ahora se hará referencia como 'I' a la imagen donde se introducirá la información, 'W' la marca de agua y 'K' la clave (es usada como la semilla del generador de números aleatorios, siendo el proceso de introducción de la información de la forma: I x K x W = I'. La figura que se muestra a continuación representa, genéricamente, la forma de introducir información en una imagen:

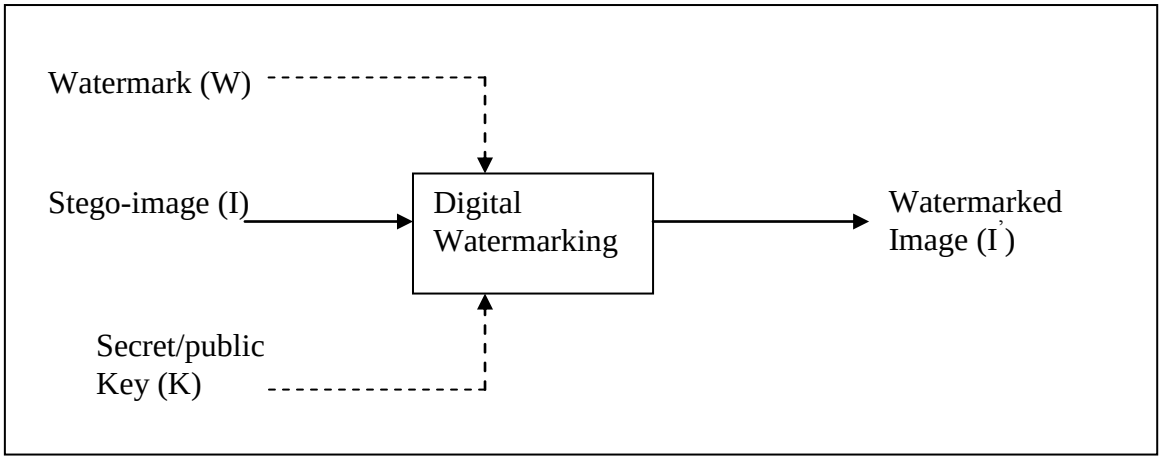

Ilustración 1 – Watermarking (1)

A continuación se muestra una figura que representa la forma de recuperar una marca de agua de una imagen:

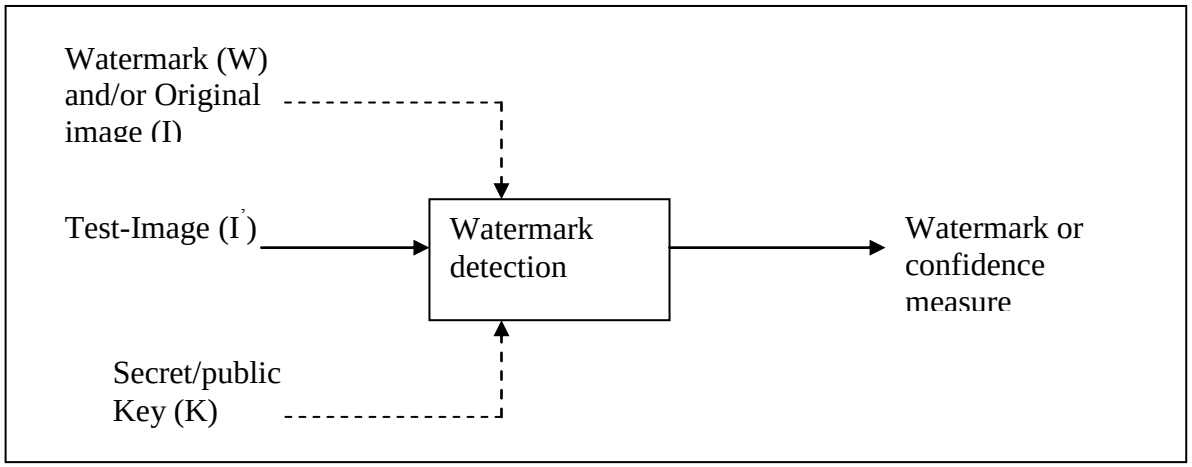

Ilustración 2 - Watermarking Detección (1)

- **Algoritmos y transformaciones:** Esta técnica oculta los datos mediante el uso de funciones matemáticas que a menudo se utilizan en algoritmos de compresión de datos. El método consiste en ocultar la información en los bits de los datos menos importantes.

- **Bit Menos Significativo (Least Significant Bit):** Esta técnica es la más usada de la esteganografía. Consiste en sustituir el bit menos significativo, de los pixeles para el caso que se trate de una imagen, o de las distintas muestras en el caso de un archivo de audio. Aunque la técnica también puede ser aplicada a archivos de video, generalmente solo se usa en imágenes, ya los archivos de audio y video se ven mucho más afectados por las distorsiones introducidas que las imágenes. A continuación se muestra un ejemplo de la introducción de la letra A en tres pixeles en formato RGB (3 bytes):

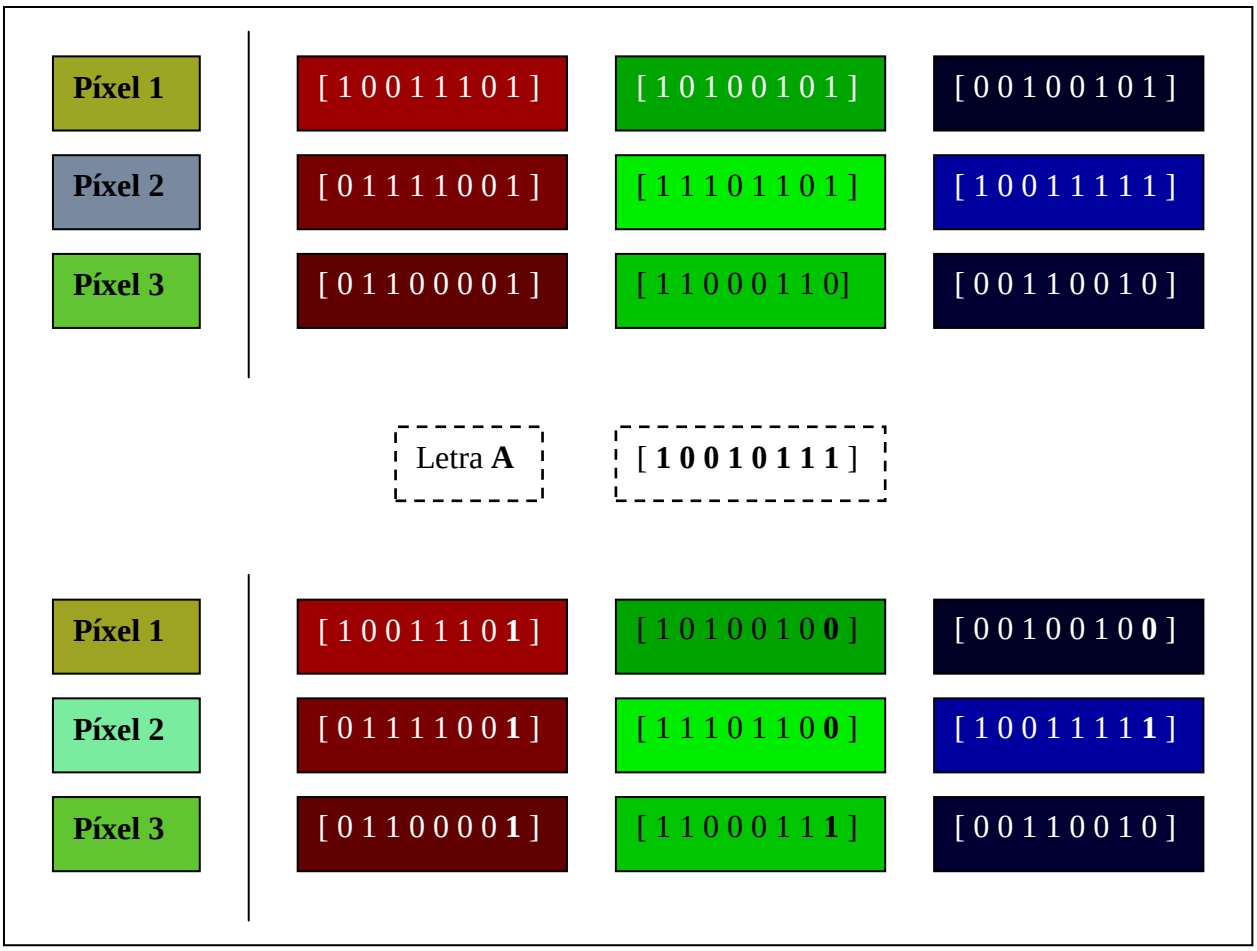

Ilustración 3 - Ejemplo LSB en Imágenes

1.4 Glosario de Términos

En este apartado se muestran las definiciones de las palabras clave del proyecto así como el significado de los acrónimos más empleados en el mismo.

1.4.1 Definiciones

- **Arquitectura Cliente-Servidor:** Una aplicación (cliente) realiza peticiones a otra (servidor) que le responde con la información solicitada.

- **Cifrar:** Transcribir en guarismos, letras o símbolos, de acuerdo con una clave, un mensaje cuyo contenido se quiere ocultar.

- **Criptografía:** Es el arte de cifrar o descifrar información mediante técnicas especiales.

- **Esteganálisis:** Es la técnica que se usa para descifrar mensajes ocultos mediante técnicas esteganográficas.

- **Esteganografía:** Es la disciplina en la que se estudian y aplican técnicas que permiten la ocultación de mensajes u objetos en otros, llamados portadores.

- **Java:** Lenguaje de programación orientada a objetos.

- **Programación orientación a objetos:** Paradigma de programación que emplea objetos y sus interacciones para diseñar aplicaciones y programas de ordenador.

- **Proxy:** Programa o dispositivo que realiza una acción en lugar de otro.

- **Servidor:** Computadora que mediante el empleo de una red provee servicios a otras.

- **Watermarking:** Es una técnica de ocultación de información que forma parte de las técnicas conocidas como esteganográficas.

- **Web:** Es un sistema de documentos de hipertexto o de hipermedios enlazados accesibles a través de Internet.

1.4.2 Acrónimos

BMP	Bitmap
GIF	Graphics Interchange Format
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
JPEG	Joint Photographic Experts Group
LSB	Least Significant Bit
MP3	MPEG-1 Audio Layer 3
MPEG	Moving Picture Experts Group
NAT	Network Address Translation

Capítulo 2
Análisis del Proyecto

En este segundo capítulo se ofrece una especificación detallada del proyecto, dividida en tres apartados. El primer apartado trata sobre el estado del arte, es decir, el marco tecnológico en el que se encuentra el proyecto. El segundo apartado muestra la especificación detallada del proyecto, especificando los requisitos de usuario y los subsistemas que lo conforman. Por último, este capítulo nos ofrece el plan de pruebas, donde se detallan las pruebas que se realizarán, tanto para la validación de la aplicación como para comprobar el rendimiento de ésta.

2.1 Estado del Arte

En este apartado se tratará el marco tecnológico en el que se encuentra enclavado este proyecto. Para ello se presentará de manera detallada las artes, técnicas y protocolos que serán empleados en el mismo.

2.1.1 Esteganografía

Como ya se explicó en el capítulo anterior, la esteganografía es la disciplina en la que se estudian y aplican técnicas que permiten ocultar mensajes u objetos dentro de otros, llamados portadores. En el caso de la esteganografía digital, la información ocultada puede ser tanto mensajes como archivos digitales, sin embargo, los portadores serán archivos de texto, archivos multimedia (imágenes, audio y video digitales), archivos ejecutables y protocolos de comunicación.

En este apartado se explicarán detalladamente las técnicas que conciernen a este proyecto, ya que se ha centrado en la eliminación de información oculta en imágenes digitales y en archivos de audio digital. Es por eso que, a continuación, se detallarán las principales técnicas para la ocultación de información en dichos tipos de archivo multimedia.

- **Esteganografía en Imágenes digitales**

  - LSB:

La técnica LSB es una de las técnicas de sustitución más antiguas y utilizadas. Está basada en el principio de que la modificación de los bits menos significativos en una imagen no produce "ruido visual" capaz de levantar sospechas de que hay información adicional dentro de la imagen, dese el punto de vista del ojo humano. Además se puede emplear un generador de números pseudo-aleatorios con el fin de hacer que los bits escogidos para ocultar el mensaje sean elegidos de manera aleatoria, mediante el empleo de una clave que serviría de semilla para el generador.

Esta técnica se puede aplicar de maneras distintas dependiendo del estegomédio elegido. Para el formato BMP se introduciría en la codificación de los pixeles de la imagen, en un fichero GIF en los índices que apuntan a la paleta de colores, en el formato JPEG a los coeficientes cuantificados DCT's, etc. La técnica se puede usar de forma secuencial, pseudo-aleatoria, o bien mezclando técnicas.

La modificación de los bits menos significativos de la imagen puede borrar la posible información oculta almacenada en la misma, haciendo esta irrecuperable.

- Outguess: (2)

El algoritmo OutGuess es un procedimiento de dos pasadas para imágenes JPEG. En la primera pasada, el algoritmo embebe el mensaje de bits usando un generador de números pseudo-aleatorios para almacenar estos en los bits menos significantes (LSB) de los coeficientes DCT's, saltando los coeficientes cuyas magnitudes son cero o la unidad. En la segunda pasada, se hacen correcciones de las magnitudes de los coeficientes para asegurar que el histograma de los DCT's de la imagen de esteganografía coinciden con los de la imagen de encubrimiento.

Anteriormente a la realización del primer paso, el algoritmo OutGuess calcula la longitud máxima que puede tener el mensaje que puede ser embebido en la imagen durante el primer paso, mientras se asegura de que luego podrá hacer las correcciones necesarias para ajustar el histograma al original durante el segundo paso. Siendo n_{01} el número de todos los DCT con valor distinto de 0 o 1 y como la mayoría de los coeficientes DCT son ceros, modificar estos introduce una gran distorsión visible en la imagen, siendo esta la razón por la que el algoritmo OutGuess no embebe ceros. Como tanto ceros como unos están en igual cantidad en los coeficientes DCT's, estos son también evitados a la hora de embeber datos. La longitud máxima del mensaje está determinada por las frecuencias de los pares LSB más desbalanceados.

Para calcular la capacidad máxima de datos embebidos (q), empezamos con la expresión para el esperado valor del histograma de los coeficientes DCT antes de embeber. Como el mecanismo de Embebido es LSB, siendo (T_c) los coeficientes DCT, la longitud máxima del mensaje quedaría de la siguiente manera:

$$Q <= \frac{2\,T_c[-2]}{T_c[-1] + T_c[-2]}$$

Ilustración 4 - Capacidad máx. Datos Embebidos (2)

Esta condición garantiza que la media habrá suficientes coeficientes con magnitud -2 que podrán ser cambiados por -1 para estar seguros de que las frecuencias entre los pares LSB (-2,-1) son preservadas después del embebido.

- Esteganografía basada en el modelo: (2)

La esteganografía basada en el modelo es un sistema de construcción de de sistemas esteganográficos basado en preservar el modelo elegido para encubrir la información en vez de en las estadísticas. El modelo de encubrimiento, c, es modelado como una variable aleatoria que puede ser dividida en dos componentes, (C_{inv},C_{emb}) donde C_{inv} es invariante respecto a la información embebida y C_{emb} es lo que se modifica durante el proceso de embeber la información. En el caso concreto de LSB, C_{inv} y C_{emb} corresponden a los siete bits más significativos y los bits menos significativos respectivamente.

La capacidad máxima de embebido es:

$$\sum \left| L(C_{inv}) \right| H\left(P(C_{emb} \left| C_{inv} = C_{inv}\right)\right)$$

Ilustración 5 - Capacidad máx. Esteg. Basada en el modelo **(2)**

- Algoritmo F5: (2)

El algoritmo F5 es un método práctico para embeber información en archivos JPEG que provee una alta capacidad esteganográfica sin sacrificar la seguridad. El algoritmo F5 comienza decrementando el valor absoluto de los coeficientes DCT en uno, preservando de esta forma el histograma de los DCT, aunque parecerá que esta ha sido comprimida usando un factor de calidad menor.

El algoritmo F5 embebe los bits de un mensaje usando un generador de números pseudo-aleatorios, usando como semilla de dicho generador una contraseña suministrada por el usuario. Los coeficientes DCT con valor cero son saltados y no usados para embeber la información.

Cuando la magnitud de un coeficiente es cambiada de 1 o -1 a 0 se le llama "shrinkage". Si "shrinkage" ocurre, se tiene que re-embeber el mismo BIT, en el siguiente coeficiente. Esto se debe a que solo se leen los bits del mensaje distintos de cero. Sin embargo, al re-embeber bits 0, se puede producir que haya muchos más ceros que unos y que esto se vea reflejado en el histograma. El

algoritmo F5 resuelve este problema redefiniendo el LSB para números negativos: LSB(x) = 1 – x mod. 2 para x<0 y LSB(x) = x % 2 para el resto.

La capacidad de embebido del algoritmo F5, siendo n el número total de coeficientes DCT y T_c el histograma de todos los DCT, es:

$$n - T_c[0] - n/64 - (T_c[-1] + T_c[1]) /2$$

Ilustración 6 - Capacidad máx. Algoritmo F5 (2)

El algoritmo F5 no puede ser detectado usando los ataques de histograma porque el embebido no está basado en el intercambio de valores. Sin embargo, es vulnerable a los ataques que usan un proceso de calibrado.

- Gifshuffle: (3)

Un conjunto de objetos n puede ser ordenado representando un numero de rango [0,n!-1], pudiendo almacenar hasta $\log_2(n)$ bits. Las imágenes en formato GIF contienen un mapa de colores de hasta 256 entradas, resultando en una capacidad máxima de almacenamiento de 1675 bits. La imagen en si está formada por un arreglo de índices apuntando a este mapa de colores. Para almacenar un mensaje en la imagen se deben seguir los siguientes pasos:

1. Dado el mensaje, de longitud m bits, y siendo n el número de colores únicos en la imagen, sí m > n-1 el mensaje es demasiado largo para dicha imagen.

2. Al principio los colores del mapa de colores están ordenados en su orden natural. Cada color RGB tiene asignado un valor (rojo * 65536 + verde * 256 + azul), y los colores están ordenados según estos valores. Cualquier color duplicado es eliminado.

3. Iterar i desde los valores 1 hasta n. Cada color n-i es colocado en la posición objetivo (m mod i), después m es dividido por i.

4. Cada color (n-1) hasta 0 es después por turnos insertado en un nuevo mapa de colores en su posición objetivo. Los colores previamente ocupantes de la posición y superiores son desplazados una posición hacia arriba.

5. Si el tamaño del mapa de colores es mayor que el número de colores únicos, el mapa de colores será acolchado con el último color del mapa de colores original.

6. Los componentes de la imagen GIF son descomprimidos, y los índices de colores son re-mapeados usando el nuevo mapa de colores. Luego se vuelve a comprimir la imagen.

En el caso de que la imagen sea un GIF animado, este proceso se deberá repetir por cada imagen que componga la imagen animada.

- **Esteganografía en Audio digital**

  - LSB: (4)

La codificación LSB es la manera más simple de introducir información en un archivo de audio. Sustituyendo el bit menos significativo de cada muestra del audio con los bits del mensaje binario, se puede llegar a almacenar una gran cantidad de información.

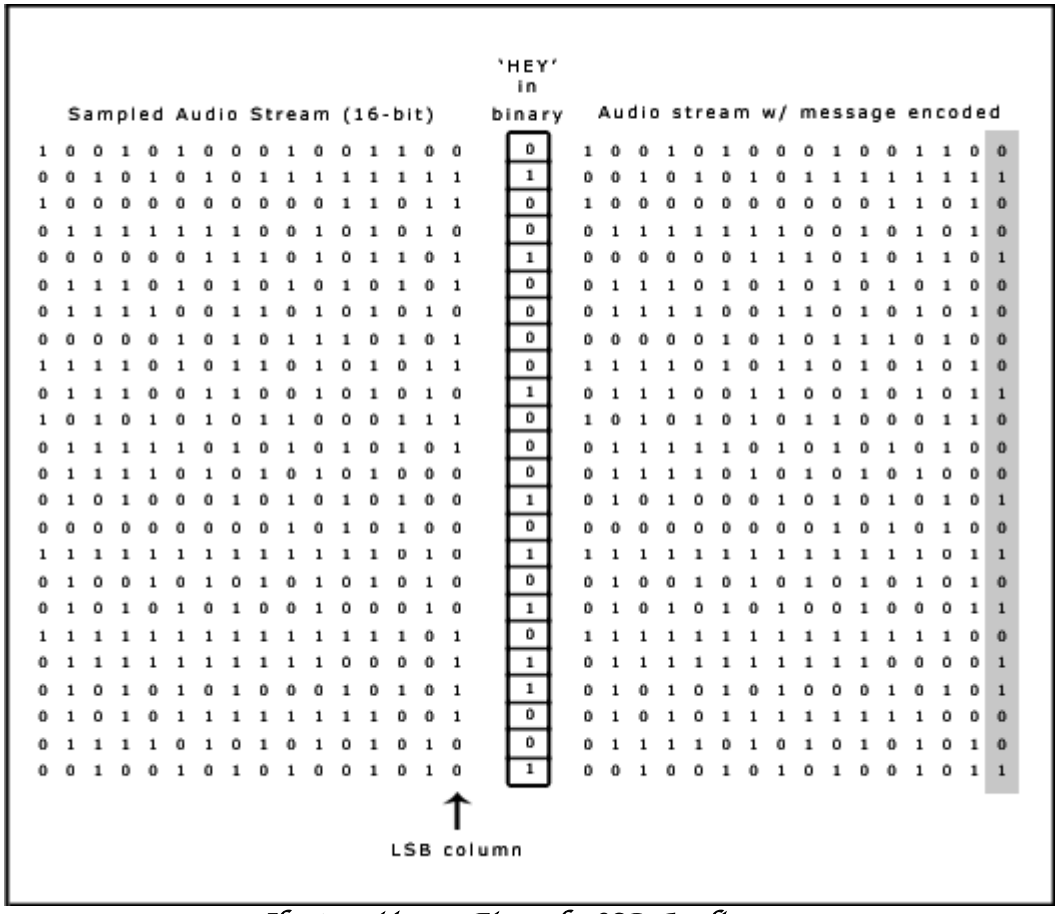

Ilustración 7 - Ejemplo LSB Audio (4)

En la codificación LSB, el ideal de transmisión de datos es 1 Kbps por 1 KHZ. Sin embargo, algunas implementaciones usan los dos bits menos significativos para reemplazarlos por dos bits del mensaje, aunque esto incrementa la cantidad de ruido en el archivo de audio. Para extraer el mensaje de un archivo de audio codificado con LSB, el receptor necesita acceder a la secuencia de ejemplo usada en el proceso de embebido. Normalmente, la longitud del mensaje es menor que el número total de muestras en el archivo de audio. Se debe elegir la forma de introducir el mensaje, a partir de que muestra se inicia el embebido, etc. Si al terminar de embeber el mensaje todavía queda muestras de audio para embeber más datos, estas pueden ser dejadas tal y como están en el audio original, pero puede crear una brecha de seguridad, al tener diferentes propiedades estadísticas la parte donde se ha introducido información respecto a la original. Una forma de solucionar este problema es una vez terminado el proceso de embebido introducir bits aleatorios igual a la longitud del mensaje en el resto del archivo de audio.

Una forma más sofisticada de embebido es el uso de un generador de números pseudo-aleatorios que use una contraseña dada por el usuario para decidir de forma aleatoria en que muestras de audio del archivo es introducida la imagen.

- Codificación de Paridad: (4)

En lugar de fragmentar una señal en muestras individuales, la codificación de paridad fragmenta la señal en regiones separadas de muestras y codifica cada bit del mensaje secreto en el bit de paridad de una región de la muestra. Si el bit de paridad de la región seleccionada no coincide con el bit que se debe codificar, el proceso altera los LSB de una de las muestras de la región.

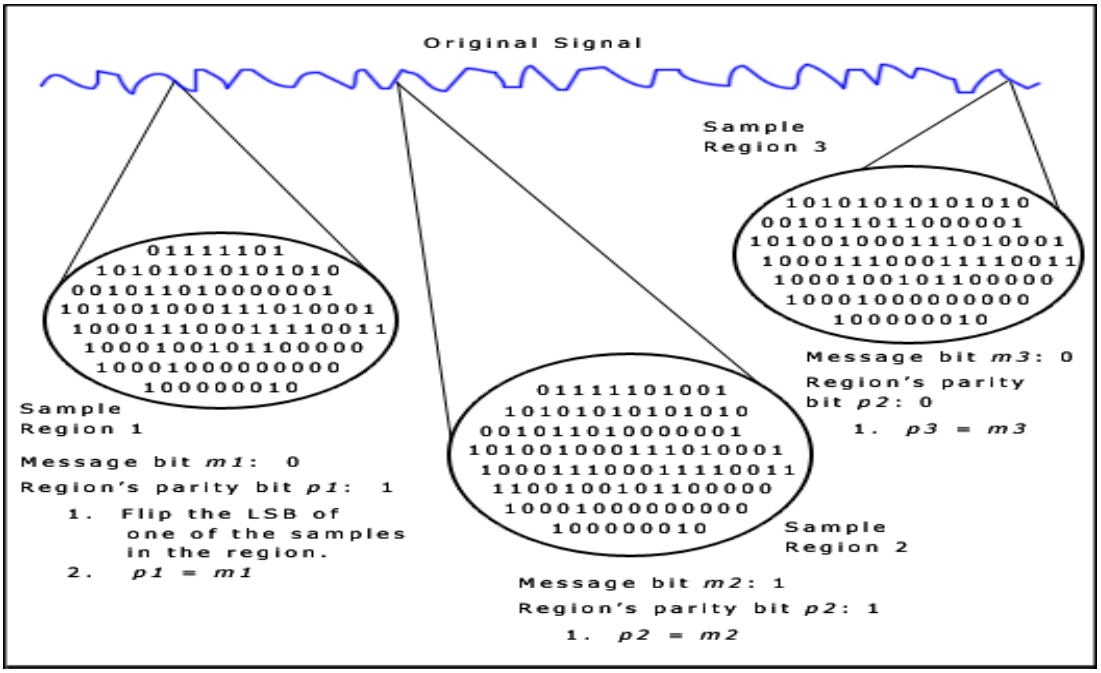

Ilustración 8 - Ejemplo Codificación de Paridad (4)

El proceso de extraer el mensaje secreto calculando los bits de paridad de las regiones usadas en el proceso de codificación. También se puede usar un generador de números pseudo-aleatorios para calcular las regiones donde introducir la información.

Existen dos desventajas principales asociadas con el uso de métodos como el LSB y la codificación de paridad. El oído humano es muy sensible y puede detectar a veces incluso el ruido más pequeño introducido en un archivo de sonido, aunque este método hace el ruido introducido casi inaudible. Ambos métodos comparten otra desventaja, no ser robustos. Si un archivo de sonido es embebido con un mensaje secreto usando cualquiera de los dos métodos, y posteriormente se volviese a embeber otro mensaje, el primer mensaje embebido se perdería. La robustez puede ser mejorada mediante el uso de técnicas de redundancia mientras se codifica el mensaje secreto, aunque dichas técnicas de redundancia reducen significativamente la tasa de transmisión de datos.

- Codificación de Fase: (4)

La codificación en fase soluciona las desventajas de los métodos de introducción de ruido, ya que se basa en el hecho de que la fase de los componentes del sonido no es tan perceptible para el oído humano como lo es el ruido. Más que introducir perturbaciones, la técnica codifica el mensaje de bits como cambios de fase en la fase del espectro de la señal digital, alcanzando una codificación inaudible en términos de la percepción de ruido en una señal.

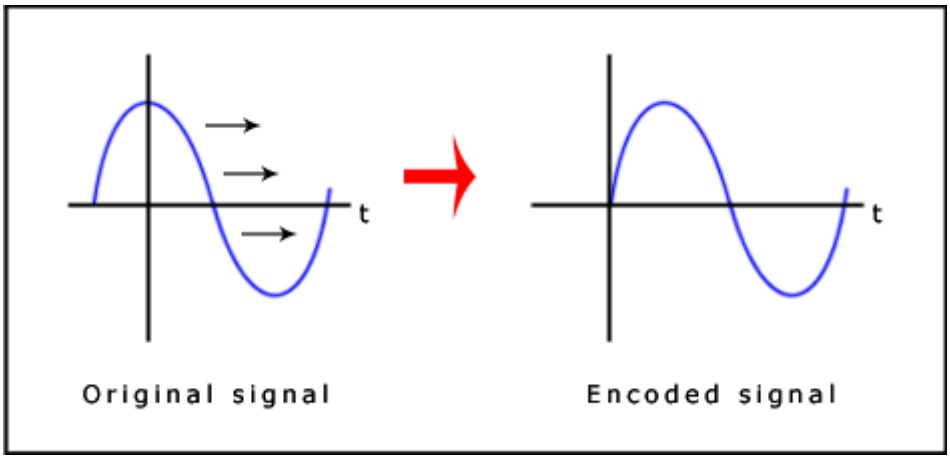

Ilustración 9 - Ejemplo Codificación en Fase (4)

La codificación en fase tiene los siguientes procedimientos:

1. La señal de sonido original es fragmentada en segmentos más pequeños cuya longitud sea igual al tamaño del mensaje a ser codificado.

2. La transformada discreta de Fourier (DFT) es aplicada a cada segmento para crear una matriz con las fases y las magnitudes de la transformada de Fourier.

3. Las fases diferenciadas entre segmentos adyacentes son calculadas.

4. Los cambios de fase entre segmentos consecutivos son fácilmente detectables. En otras palabras, las fases absolutas de los segmentos pueden ser cambiadas pero las relativas diferentes fases entre segmentos adyacentes han de ser preservadas. El mensaje secreto solo es insertada en el vector de la fase de la primera señal del segmento de la siguiente forma:

$$\text{Nueva\_fase} = \begin{cases} \prod / 2 & \text{Si el BIT} = 0 \\ -\prod / 2 & \text{Si el BIT} = 1 \end{cases}$$

Ilustración 10 - Codificación en Fase (4)

5. Se crea una nueva matriz de fases usando las nuevas fases de los primeros segmentos y las fases originales del resto.

6. Usando la nueva matriz de fases y la matriz de magnitudes originales, la señal de sonido es reconstruida aplicando el inverso DFT y luego concatenando los segmentos de sonido.

Para extraer el mensaje de un archivo de audio, el receptor debe conocer la longitud del segmento. El receptor puede luego usar el DFT para conseguir las fases y extraer la información.

Una desventaja asociada con la codificación en fase es la baja tasa de transmisión debido a que el mensaje secreto codificado solo está segmentado en el primer segmento de la señal. Esto puede ser solucionado incrementando la longitud de los segmentos de la señal. Sin embargo, esto puede cambiar las relaciones de fase entre la frecuencia de cada componente del segmento de manera drástica, haciendo más sencillo de detectar el mensaje codificado. Por estas razones, este método es usado cuando solo hay una pequeña cantidad de datos, como en el caso de las marcas de agua.

- Extensión del Espectro: (4)

El método de extensión del espectro (SS) trata de extender la información secreta a lo largo del espectro de frecuencia de la señal de audio tanto como se pueda. Este sistema es análogo a un sistema LSB que codifique el mensaje aleatoriamente a lo largo de todo el archivo de sonido. Sin embargo, en contra de lo que ocurre con el LSB, el método SS extiende el mensaje secreto por el espectro de frecuencia del archivo de sonido, usando un código que es independiente de la señal actual. Como resultado, la señal final ocupa un ancho de banda que excede lo que en realidad es necesario para la transmisión.

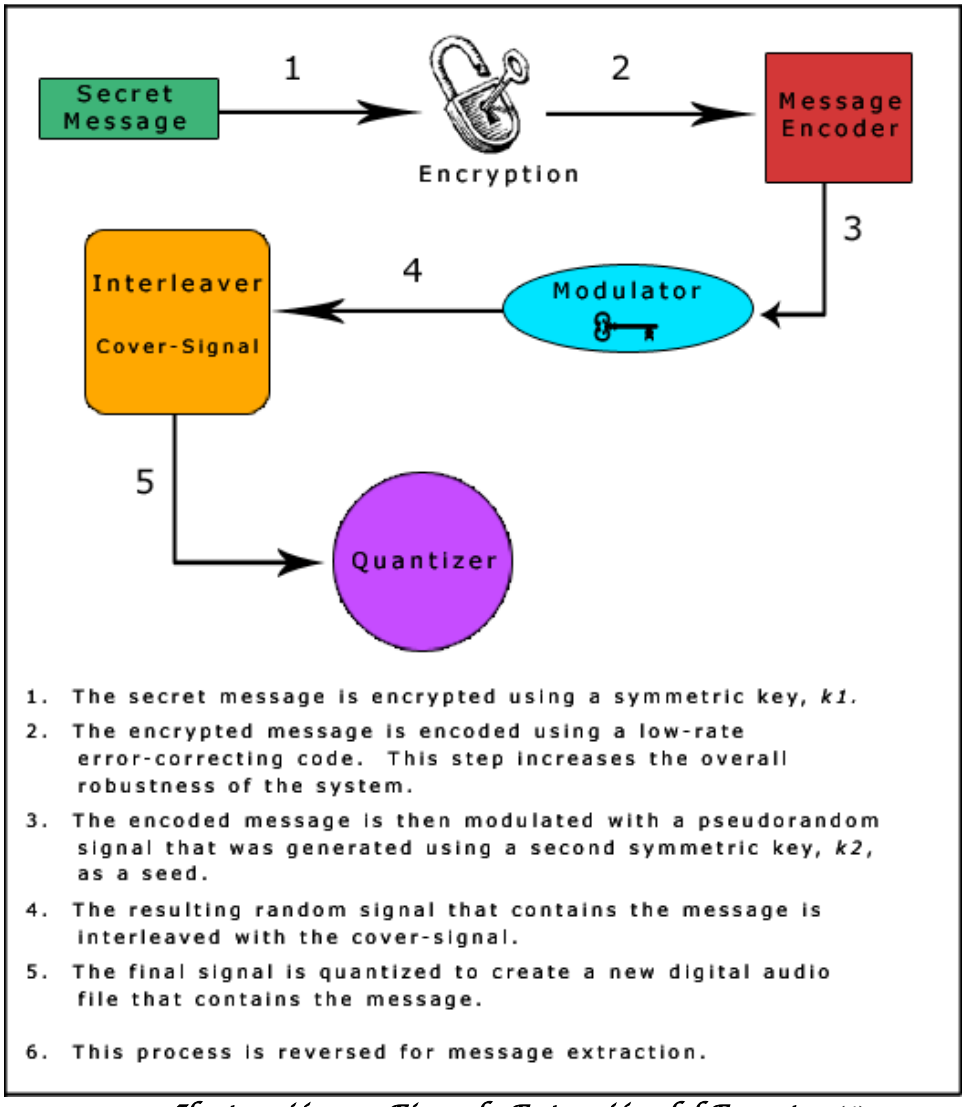

Ilustración 11 - Ejemplo Extensión del Espectro (4)

Existen dos versiones del SS que pueden ser usadas en esteganografía en archivos de audio, el esquema de secuencia directa y el esquema de salto de frecuencia. El primer método, el mensaje es esparcido por una constante llamada ratio de "paso" y luego modulado por una señal pseudo-aleatoria.

Luego es mezclado con la señal de encubrimiento. En el segundo método, el espectro de frecuencia del archivo de audio es alterado de tal forma que este salte rápidamente entre frecuencias.

El método SS es mejor en algunas áreas que el LSB, la codificación de paridad y la codificación de fase, ofreciendo una ratio de transmisión de datos moderado mientras también mantiene un alto nivel de robustez contra las técnicas de eliminación. Sin embargo, el método SS comparte desventajas con el LSB y la codificación de paridad en que introduce ruido que puede ser oído en el archivo de sonido.

- Ocultamiento de Eco: (4)

En ocultamiento de eco, la información es embebida en un archivo de audio introduciendo eco en la señal. Al igual que el método de extensión de espectro, este también tiene las ventajas que permiten un alto ratio de transmisión de datos y provee una robustez superior en comparación con los métodos de adición de ruido.

Para esconder la información de manera exitosa se necesitan tres parámetros del eco, la amplitud, la tasa de decaimiento y el Offset de la señal original. Estos parámetros se ajustan por debajo del umbral de sonido del oído humano. Además, el Offset se varía para representar un mensaje binario a codificar. Un valor del Offset representa un 1 binario, y un segundo Offset representa un 0 binario.

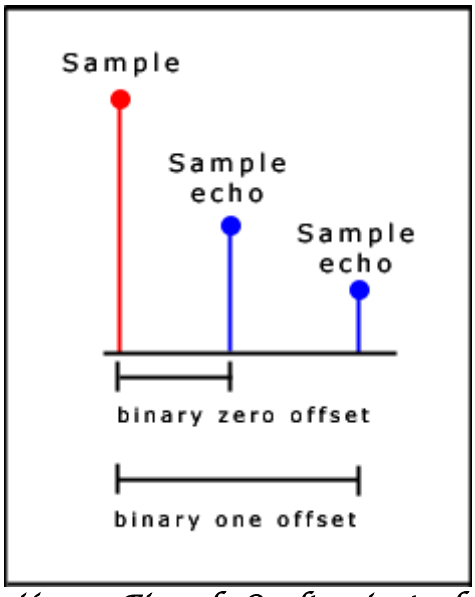

Ilustración 12 - Ejemplo Ocultamiento de Eco (4)

La señal original es despedazada en bloques para permitir codificar más de un bit a lo largo de ella, siendo unida de nuevo para crear la señal final una vez se han codificado todos los bits.

Usando la técnica de ocultamiento de eco, puede resultar que el producto sea una mezcla de ecos en la señal claramente detectable, incrementando el riesgo de detección. Una segunda implementación de la técnica soluciona el problema. Primero se crea una señal de eco a partir de la original, usando los valores cero del Offset. Luego se crea otra señal usando los valores uno del Offset. Para combinar las dos señales de eco para conseguir la codificación final, dos señales mezcladoras tienen que ser usadas. Las señales mezcladoras tienen el valor 0 o 1 dependiendo de qué BIT deba ser codificado en el bloque.

Para extraer el mensaje de la señal esteganografiada, el receptor debe poder romper la señal en la misma secuencia de bloques usada durante el proceso de codificación. Luego la función auto-correlación de la señal *espectrum* (la *espectrum* es la transformada Fourier de el espectro de frecuencias de la señal) puede ser usada para descodificar el mensaje, (4) porque revela cada Offset del eco, permitiendo que el mensaje pueda ser reconstruido.

2.1.2 Herramientas Esteganográficas

En este apartado se hará una breve descripción de las distintas herramientas esteganográficas que se pueden utilizar para la ocultación de mensajes en imágenes y en audio digital.

- **Herramientas que usan imágenes:**

  - BlindSide: (5)

Aplicación esteganográfica que permite ocultar un archivo en una imagen BMP. Permite también cifrar los datos mediante el uso de una contraseña, previamente aportada por el usuario.

  - JPHide and JPSeek: (6)

Esta aplicación permite ocultar un archivo en una imagen JPEG. No permite el cifrado de la información, ocultando dicha información en claro en la imagen.

- Gifshuffle: (3)

Oculta la información en la paleta de colores de una imagen GIF mediante el reordenado de la misma. Emplea el algoritmo de cifrado ICE, creado por el mismo autor, para cifrar la información antes de embeberla en la imagen.

- MP3Stego: (7)

Comprime archivos de audio digital en formato WMA a formato MP3, introduciendo la información que se desea embeber durante el proceso. Oculta la información en los bits de paridad de las distintas regiones del audio.

2.1.3 Esteganálisis

El Esteganálisis es la ciencia de detectar mensajes ocultos mediante el uso de la esteganografía, siendo análogo al criptoanálisis aplicado a la criptografía. Generalmente, se clasifica el esteganálisis en dos categorías en función del objetivo que se desea alcanzar.

El esteganálisis pasivo se desea detectar la presencia o la no presencia de información oculta en un archivo digital. En el caso de existir información oculta, la metas principales del esteganálisis son averiguar el algoritmo con el que se ha embebido la información, la longitud del mensaje embebido, estimar su localización, estimar la clave y otros parámetros usados por el algoritmo, y finalmente, extraer el mensaje embebido, algo que debe considerarse como un gran logro. Las técnicas básicas para la obtención de estas informaciones son los análisis estadísticos. También facilita la labor la tenencia de archivos no modificados del mismo tipo que han sido usados para el embebido del mensaje (por ejemplo, fotos de la misma cámara digital).

El guardián activo monitoriza la comunicación realizada a través de un canal de comunicación, introduciendo pequeñas modificaciones en los datos transmitidos, eliminando los mensajes ocultos de un portador específico. Este guardián es el sistema de protección desarrollado en este proyecto, ya que la intención es eliminar la información esteganográfica enviada mediante el empleo del proxy y de los filtros esteganográficos. A continuación, se muestran algunos de los algoritmos existentes capaces de eliminar la información de los algoritmos y programas esteganográficos citados en los apartados anteriores:

- Eliminación de LSB

El algoritmo LSB introduce la información que se desea en los Bits menos significativo del archivo, ocurriendo que la eliminación de dicha información sea factible solo con alterar los bits menos significativos. Como se desconoce tanto la longitud del mensaje como en que bits ha sido ocultado, para asegurarse la eliminación del mismo se debe cambiar todos los bits menos significativos del archivo por 0 ó 1 de manera aleatoria, consiguiendo que no se pueda recuperar el mensaje oculto en el destino.

Está técnica puede ser usada tanto en archivos de audio como en archivos de imágenes. En el caso de archivos de audio se debe modificar los bits menos significativos de cada muestra. Para los archivos de imagen, se debe modificar los bits menos significativos de los BMP, la paleta de colores en el caso de los archivos GIF o los coeficientes DCT's en el caso de los archivos JPEG.

Se debe tener en cuenta que está técnica puede producir ruido audible por el ser humano en los archivos de audio, debiéndose valorar su utilización en cada caso concreto dependiendo de si se puede permitir la pérdida de calidad. También se puede utilizar esta técnica con el resto de bits de cada byte de la imagen, pero en ese caso se debe tener en cuenta que el ruido de la imagen aumentaría en exceso.

Este ataque puede ser usado contra las técnicas de esteganografía LSB tanto para audio como para imágenes, para la técnica codificación de paridad de archivos de audio, y para las técnicas Outguess, esteganografía basada en el modelo y el algoritmo F5.

- Ataque a la paleta de colores de GIF

En los archivos GIF el mensaje secreto puede ser ocultado en la paleta de colores, ordenando esta de forma que represente el mensaje a enviar. Para la eliminación de dicho mensaje oculto, se debe reordenar la paleta de colores del archivo GIF, evitando así la recuperación de este. Para asegurar la eliminación del mensaje, la reordenación de los colores de la paleta se deberá hacer de forma aleatoria. Una vez reordenada la paleta, se deberá reasignar a cada pixel de la imagen su color correspondiente en la nueva paleta.

Está técnica solo es válida para archivos GIF a los que les hayan embebido un mensaje secreto mediante la técnica gifshuflle.

- Compresión JPEG

Realizar una compresión de la imagen usando el algoritmo de compresión del formato JPEG puede permitir la eliminación de un mensaje oculto. Este ataque es soportado por la mayoría de los sistemas, siempre hasta cierto nivel de compresión. Está técnica solo es válida para archivos de imagen.

- Transformaciones geométrica

Se puede borrar mensajes ocultos en imágenes aplicando diversas transformaciones geométricas a las mismas. Para que estas transformaciones sean eficaces deben ser generalizadas, es decir, se deben efectuar varias consecutivas para aumentar las probabilidades de éxito del proceso. Una de estas opciones es la combinación de redimensionamientos no uniformes, volteos (Horizontales para que la imagen no pierda el sentido) y rotaciones (de muy pocos grados). Estas transformaciones no pueden ser muy grandes para no provocar una distorsión del sentido de la imagen.

Estos ataques pueden ser realizados sobre todo tipo de imágenes a excepción de las que estén en formato GIF, ya que en este caso la información se oculta en la paleta de colores de la misma, y no en sus pixeles.

2.1.4Protocolo HTTP

HTTP *(8)(9)* es un protocolo Cliente-Servidor para el intercambio de información entre clientes Web y servidores HTTP utilizando sencillas operaciones de solicitud-respuesta. Las comunicaciones se realizan mediante TCP/IP a través del puerto 80. El propósito principal del protocolo es permitir la transferencia de archivos (principalmente HTML) entre un navegador y un servidor Web localizado mediante una cadena de caracteres llamada URL. Los encabezados de los mensajes describen el contenido del mismo mediante la codificación MIME.

Las comunicaciones entre el cliente y el servidor se llevan a cabo en dos etapas. En la primera etapa, el cliente Web realiza una solicitud HTTP. En la segunda etapa el servidor procesa dicha solicitud y después envía una respuesta al cliente.

Una solicitud HTTP está formada por un conjunto de líneas que el navegador envía al servidor compuesto por una línea de solicitud formada por el comando, la dirección URL y la versión del protocolo utilizada por el cliente, todo ello separado por espacios y la dirección del host. Las siguientes líneas los campos del encabezado de solicitud. Son un conjunto de líneas opcionales que

permiten aportar información adicional sobre la solicitud, el cliente (navegador, sistema operativo, etc.). Cada una de estas líneas está formada por un nombre que describe el tipo de encabezado, seguido de dos puntos (:) y el valor del encabezado. El último conjunto de líneas que forman la solicitud es el cuerpo de la solicitud, donde deben estar separadas de las líneas precedentes por una línea en blanco y que permiten que se envíen datos por un comando POST durante la transmisión de datos al servidor.

A continuación se muestra en sendas tablas los distintos comandos y encabezados del protocolo HTTP:

Comando	Descripción
GET	Solicita el recurso ubicado en la URL especificada. Existe también los GET "condicionales" si son acompañados por "If-Modified-Since", "If-Unmodified-Since", "If-Match", "IF-None-Match" o "if-Range" en el campo cabecera.
HEAD	Solicita el encabezado del recurso ubicado en la URL especificada, devolviendo la meta-información sobre la URL y no el cuerpo del mensaje.
POST	Envía datos al programa ubicado en la URL especificada.
PUT	Envía datos a la URL especificada.
DELETE	Borra el recurso ubicado en la URL especificada.
TRACE	Es usado para invocar un programa remoto la respuesta de un mensaje.
CONNECT	Usado con un Proxy permite cambiar para ser un túnel.

Tabla 1 - Comandos HTTP (8) (9)

Encabezados	Descripción
Accept	Tipo de contenido aceptado por el navegador (Tipos de MIME).
Accept-Charset	Juego de caracteres que el navegador espera.
Accept-Encoding	Codificación de datos que el navegador acepta.
Accept-Language	Idioma que el navegador espera.
Accept-Ranges	Permite al servidor indicar la aceptación de un rango de respuestas de una fuente.
Authorization	Identificación del navegador en el servidor.
Cache-Control	Usado para especificar directivas que se deben obedecer por todos los mecanismos de cache.
Connection	Tipo de conexión que el usuario prefiere.
Cookie	Una cookie http previamente enviada al servidor con Set-Cookie.
Content-Length	Extensión del cuerpo de la solicitud.
Content-Type	El tipo MIME del cuerpo de la respuesta.
Date	Fecha de envío del mensaje.

Expect	Indica que comportamiento del servidor es requerido por el cliente.
From	La dirección de correo electrónico del usuario que hace la respuesta.
Host	El dominio del servidor.
If-Match	Solo realizar la acción si la entidad aportada por el cliente cuadra con la del servidor.
If-Modified-Since	Permite un "304 Not Modified" en respuesta si el contenido no ha sido modificado.
If-None-Match	Permite un "304 Not Modified" en respuesta si el contenido no ha sido modificado.
If-Range	Si la entidad no ha sido modificada, envía la parte o partes que he perdido, sino envía una entidad nueva.
If-Unmodified-Since	Solo envía la respuesta si la entidad no ha sido modificada desde un tiempo específico.
Max-Forwards	Numero límite de veces que el mensaje puede ser enviado a través de varios Proxy.
Pragma	Implementa cabeceras específicas que pueden tener varios efectos.
Proxy-Authorization	Credenciales para conectarse a un Proxy.
Range	Petición de una parte de una entidad.
Referer	Es la dirección de la página Web desde la cual he entrado a la actual.
TE	La codificación de las transferencias que el usuario tiene la voluntad de aceptar.
Upgrade	Pregunta al servidor para actualizar otro protocolo.
User-Agent	Una cadena del usuario.
Via	Informa al servidor de los proxys atravesados por la respuesta que fue enviada.
Warn	Un aviso general sobre posibles problemas con el cuerpo del la entidad.

Tabla 2 - Encabezados Solicitud HTTP (8) (9)

Una respuesta HTTP, al igual que una solicitud, está formada por un conjunto de líneas, que al contrario que las solicitudes, son enviadas del servidor al navegador. Está formada por una línea de estado que especifica la versión del protocolo utilizada, el código de estado y el significado del código. Después viene seguida por otro conjunto de líneas opcionales que aportan información sobre la respuesta, el protocolo, o ambas. Cada línea está formada por un nombre que califica el tipo de encabezado, seguido por dos puntos (':') y por el valor del encabezado. A continuación iría el cuerpo de la respuesta que contendría el documento solicitado.

Existen dos tipos de recursos que un servidor Web provee a los clientes, recursos estáticos y recursos dinámicos. Los recursos estáticos son archivos como los documentos HTML, documentos de texto, imágenes y audio. Cuando

un recurso estático es solicitado, el servidor los carga del sistema de ficheros y los envía al cliente encapsulados en una respuesta http. Habitualmente la información estática es cacheada para una mayor eficiencia ya que no suele variar. (9)

Los recursos dinámicos son generados como el resultado de una petición a una URL, que representa un programa o una base de datos en vez de un archivo estático. En el caso de un programa, el servidor lanza una solicitud al programa, enlazando la información del cliente a este, y envía la respuesta del programa al cliente como una respuesta a la solicitud. Para los accesos a bases de datos, la petición del cliente es trasladada a un formulario de la base de datos y el resultado es formateado y devuelto al cliente. (9)

Encabezados	Descripción
Accept-Ranges	Permite al servidor indicar la aceptación de un rango de respuestas de una fuente.
Age	El tiempo en segundos que ha estado el objeto en un Proxy con cache.
Allow	Acciones validas para un recurso específico.
Cache-Control	Usado para especificar directivas que se deben obedecer por todos los mecanismos de cache.
Content-Encoding	Tipo de codificación para el cuerpo de la solicitud.
Content-Language	Tipo de idioma en el cuerpo de la solicitud.
Content-Length	Extensión del cuerpo de la solicitud.
Content-Location	Una localización alternativa para la respuesta de datos.
Content-Dispositon	La oportunidad de empezar un dialogo “File Download” para un tipo MIME conocido.
Content-MD5	Un sistema de cifrado en base 64 binario MD5 añadido a contenido de la respuesta.
Content-Range	Donde dentro del cuerpo completo del mensaje una parte de este debe ir.
Content-Type	Tipo de contenido del cuerpo de la solicitud (Tipos de MIME).
Date	Fecha en que comienza la trasferencia de datos.
Etag	Un identificador para la versión específica de una fuente.
Expires	Fecha límite de uso de los datos.
Last-Modified	Fecha de la última modificación del objeto solicitado.
Location	Redireccionamiento a una nueva dirección URL asociada con el documento.
Pragma	Implementa cabeceras específicas que pueden tener varios efectos.
Proxy-Authenticate	Respuesta de autentificación para acceder al Proxy.
Refresh	Usado en redirecciones o cuando un nuevo recurso es creado. Esta recarga redirige tras 5 segundos.
Retry-After	Si una entidad esta temporalmente no disponible, instruye al cliente para reintentarlo de nuevo tras un tiempo

	específico.
Server	Características del servidor que envió la respuesta.
Set-Cookie	Una cookie http.
Trailer	Indica que los campos presentados en el cuerpo del mensaje están cifrados.
Transfer-Encoding	La forma de cifrar para transferir seguramente la entidad al usuario.
Vary	Indica como emparejar futuras peticiones para decidir si la respuesta cacheada puede ser usada o si se debe solicitar una nueva al servidor de origen.
Via	Informa al cliente de los proxys que ha atravesado la respuesta
Warning	Un aviso general sobre posibles problemas con el cuerpo del la entidad.
WWW-Authenticate	Indica la autentificación que debe ser usada para acceder a la entidad requerida.
Keep-alive	Tiempo de espera en escucha de más peticiones si así se requiere.

Tabla 3 - Encabezados Respuesta HTTP (8) (9)

Los códigos de respuesta que se ven cuando el navegador no puede mostrar la página solicitada están formados por tres dígitos, el primero índica el estado y los dos siguientes explican la naturaleza exacta del error. A continuación se pueden observar dichos códigos y sus descripciones:

Código	Mensaje	Descripción
10x	**Mensaje de información**	**Estos códigos no se utilizan en la versión 1.0 del protocolo**
20x	**Éxito**	**Estos códigos indican la correcta ejecución de la transacción**
200	OK	La solicitud se llevó a cabo de manera correcta
201	CREATED	Sigue a un comando POST e indica el éxito, la parte restante del cuerpo indica la dirección URL donde se ubicará el documento creado recientemente.
202	ACCEPTED	La solicitud ha sido aceptada, pero el procedimiento que sigue no se ha llevado a cabo
203	PARTIAL INFORMATION	Cuando se recibe este código en respuesta a un comando de GET indica que la respuesta no está completa.
204	NO RESPONSE	El servidor ha recibido la solicitud, pero no hay información de respuesta
205	RESET CONTENT	El servidor le indica al navegador que borre el contenido en los campos de un formulario
206	PARTIAL CONTENT	Es una respuesta a una solicitud que consiste en el encabezado *range*. El servidor debe indicar el encabezado *content-Range*

30x	Redirección	Estos códigos indican que el recurso ya no se encuentra en la ubicación especificada
301	MOVED	Los datos solicitados han sido transferidos a una nueva dirección
302	FOUND	Los datos solicitados se encuentran en una nueva dirección URL, pero, no obstante, pueden haber sido trasladados
303	METHOD	Significa que el cliente debe intentarlo con una nueva dirección; es preferible que intente con otro método en vez de GET
304	NOT MODIFIED	Si el cliente llevó a cabo un comando GET condicional (con la solicitud relativa a si el documento ha sido modificado desde la última vez) y el documento no ha sido modificado, este código se envía como respuesta.
40x	**Error debido al cliente**	**Estos códigos indican que la solicitud es incorrecta**
400	BAD REQUEST	La sintaxis de la solicitud se encuentra formulada de manera errónea o es imposible de responder
401	UNAUTHORIZED	Los parámetros del mensaje aportan las especificaciones de formularios de autorización que se admiten. El cliente debe reformular la solicitud con los datos de autorización correctos
402	PAYMENT REQUIRED	El cliente debe reformular la solicitud con los datos de pago correctos
403	FORBIDDEN	El acceso al recurso simplemente se deniega
404	NOT FOUND	Un clásico. El servidor no halló nada en la dirección especificada. Se ha abandonado sin dejar una dirección para redireccionar... :)
50x	**Error debido al servidor**	**Estos códigos indican que existe un error interno en el servidor**
500	INTERNAL ERROR	El servidor encontró una condición inesperada que le impide seguir con la solicitud (una de esas cosas que les suceden a los servidores...)
501	NOT IMPLEMENTED	El servidor no admite el servicio solicitado (no puede saberlo todo...)
502	BAD GATEWAY	El servidor que actúa como una puerta de enlace o proxy ha recibido una respuesta no válida del servidor al que intenta acceder
503	SERVICE UNAVAILABLE	El servidor no puede responder en ese momento debido a que se encuentra congestionado (todas las líneas de comunicación se encuentran congestionadas, inténtelo de nuevo más adelante)
504	GATEWAY TIMEOUT	La respuesta del servidor ha llevado demasiado tiempo en relación al tiempo de espera que la puerta de enlace podía admitir (excedió el tiempo asignado...)

Tabla 4 - Códigos Respuesta HTTP (8) (9)

2.1.5 Proxy

Un servidor proxy es un computador con un software específico capaz de mediar entre las conexiones de red que un cliente hace con un servidor. Funciona haciendo que tanto el cliente como el servidor se conecten entre sí a través de el proxy, es decir, usan al proxy como intermediario de todas las comunicaciones que se establecen entre ambos. Concretamente, cuando un cliente quiere acceder a un recurso de un servidor, este no se lo pide directamente a este, sino que la petición se realiza al proxy. El proxy se debe encargar de procesar la solicitud y efectuar el papel de cliente con respecto al proxy, obteniendo de esta manera la información deseada por el cliente, al cual se le remite la misma tras ser obtenida por el proxy.

Existen varios tipos de proxy, según el tipo de servicio que presten al cliente. Algunos de los tipos de proxy existentes son los siguientes:

- **Filtro de contenidos:** Se emplean para filtrar contenidos que no se desean que sean accedidos. Por ejemplo, para filtrar contenido para adultos en centros educativos.

- **Control de accesos a contenidos:** Implementa una estrategia de control de acceso para un conjunto de servidores y recursos Web. También sirve como un medio para facilitar auditorias.

- **Cortafuegos de seguridad:** Permite restricciones con los protocolos a nivel de aplicación tanto para el flujo de entrada como para el de salida, sirviendo como un punto de seguridad para la red.

- **Cache de Web:** Sirve para mantener almacenados los archivos que han sido demandados para futuras peticiones, disminuyendo los accesos a Internet y aumentando la velocidad de conexión.

- **Reverse-Proxy:** También conocido como "Surrogate", sirve para enmascarar los servidores Web. El proxy recibe las peticiones como si se tratase de un servidor, pero al contrario que un servidor, puede realizar comunicaciones con otros servidores para buscar el recurso solicitado.

- **Enrutador de contenido:** Se utiliza para encaminar las peticiones en función del tráfico de la red y el tipo de contenido solicitado.

- **Transcoder:** Pueden modificar el cuerpo de los mensajes antes de enviarlos. Se usan para traducciones de lenguajes, formatos de datos, etc.

- **Anónimo:** Su utilidad reside en permitir un aumento de la privacidad permitiendo ser anónimo mediante la eliminación de cualquier característica distintiva como la dirección IP del cliente, algunas cabeceras de identificación como *From* y *Referer*, cookies, etc.

El uso de servidores proxy puede tener muchas ventajas, pero también algunos inconvenientes. Tanto las ventajas y los inconvenientes depende del tipo de servidor proxy que se utilice. Entre las ventajas de su uso se encuentran:

- En el caso de un proxy cache, el ahorro de tráfico y la descarga de trabajo de los servidores Web, al hacerse las peticiones al proxy y no al servidor directamente. Aumenta la velocidad gracias a la cache del proxy, que puede evitar peticiones repetidas a un servidor Web para un mismo recurso.

- Filtrar contenidos que no se desean que sean accedidos, basándose en los criterios previamente establecidos, creando una restricción cuando sea necesario.

- Modificar contenido siguiendo una función de filtrado, con el objetivo de proteger la privacidad de los usuarios, eliminando la información que no desea ser enviada basándose en criterios de seguridad.

Las desventajas del uso de proxy son las siguientes:

- Usando un proxy cache, los recursos solicitados pueden no estar actualizados si han sido modificados y el proxy cache no ha actualizado la información.

- Acceder a través de un Proxy a Internet puede impedir realizar operaciones avanzadas a través del uso de algunos puertos o protocolos.

- Los clientes pueden ver violadas su intimidad al almacenarse información a la que estos acceden o datos personales y privados de los mismos.

Otra diferenciación de tipo de proxy es según la implementación del mismo. Se llama proxy transparente a aquel que permite es aquel que para ser utilizado debe ser configurado primero manualmente, haciendo posible evadir

el mismo con el simple hecho de cambiar la configuración de este. Existen también los proxys NAT. La traducción de direcciones de red, NAT, también conocida como enmascaramiento de IP, obliga a compartir una misma dirección IP pública, obligando a todas las comunicaciones de red a pasar por el proxy de manera obligatoria. Por último están los proxys abiertos, que permiten conectarse al mismo tanto a miembros de su propia red como a miembros externos a la red.

2.1.6 Herramientas Símilares

En la actualidad existen muchos proxys que permiten una navegación segura y privada por la red. Dichos proxys pueden ser aplicaciones, que solo necesitan ser instaladas para empezar a usarlas, o páginas Web que ofrecen proxys abiertos, permitiendo que cualquier persona pueda conectarse a este y navegar a través de él (por ejemplo, http://proxyanonimo.es/ (11)).

A las aplicaciones esteganográficas, mencionadas anteriormente en este mismo capítulo, se le pueden añadir otras aplicaciones dedicadas a labores de esteganálisis. Estas aplicaciones averiguan si existe información embebida dentro de algún portador, en cuyo caso intentan extraerla (por ejemplo, StegAlizerSS (12)). Sin embargo, estás herramientas suelen ofrecer un servicio de esteganálisis activo, intentando recuperar la información embebida, mientras que este proyecto tiene el enfoque contrario, la eliminación de la información y no su recuperación.

Aunque en la actualidad existen numerosas aplicaciones de seguridad, tales como cortafuegos, proxys para filtrar contenidos, etc., no existe ningún proxy esteganográfico como el que se está desarrollando en este proyecto, ya que, normalmente, las aplicaciones de esteganálisis que existen son programas capaces de analizar ficheros en busca de información oculta, pero no suelen ejecutarse para filtrar el contenido que entra y sale de un red, con el fin de eliminar cualquier información oculta dentro de ese contenido introducida por medios esteganográficos.

2.2 Descripción Detallada

En este apartado se ofrecerá una descripción detallada de la aplicación, incluyendo los requisitos de usuario y los subsistemas que conforman la misma. Como ya se explicó anteriormente, la aplicación está formada por un proxy HTTP y una serie de filtros esteganográficos. La labor del proxy HTTP será la de cualquier proxy, interceptar las comunicaciones de red que se realicen mediante el protocolo HTTP y reenviar estás a su destino, haciendo de intermediario entre el cliente y el servidor Web. Los filtros esteganográficos deberán ser capaces de filtrar toda la información que atraviese el proxy, eliminando cualquier información que haya sido ocultada en los datos enviados por medios esteganográficos.

El proxy estará implementado mediante el empleo de hilos, permitiendo así que cada comunicación HTTP que se realice sea atendida por un hilo independiente. También se hará uso de socket para el establecimiento de la comunicación. Éstos se conectarán con el cliente para recibir la solicitud HTTP y para enviar la respuesta HTTP. También se conectaran con los servidores para retransmitir a éstos las solicitudes del cliente y obtener las respuestas a las mismas. El proxy se encargará de filtrar tanto las solicitudes como las respuestas antes de retransmitirlas a sus oportunos receptores

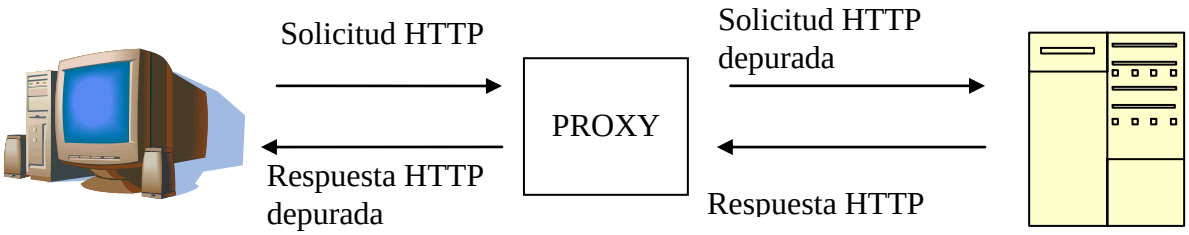

Ilustración 13 - Ejemplo Proxy

El proxy se encargará de invocar los distintos filtros esteganográficos cada vez que una solicitud o una respuesta envíen información a través del proxy. Los filtros esteganográficos deberán eliminar la información oculta en los distintos archivos. Para ello, se deberá crear un filtro específico para cada tipo de dato, tales como imágenes, audio, etc. Los filtros devolverán la misma imagen, pero sin información oculta en el caso de que previamente la hubiese, procurando evitar modificaciones sobre los datos que puedan provocar que se note claramente que estos han sido manipulados.

La aplicación trabajará sin la necesidad de un usuario más allá de su arranque, es decir, la aplicación será capaz por sí sola de interceptar las comunicaciones HTTP que se produzcan, pasando los filtros a la información enviada cuando sea necesario. El usuario solo deberá interactuar con la aplicación para arrancar la misma, siendo la manipulación de los datos un proceso automático.

2.2.1 Especificación de Requisitos de Usuario

Los requisitos de usuario especifican las funcionalidades que tendrá el sistema. Existen dos tipos de requisitos, los funcionales, que describen las distintas funciones que el sistema deberá tener, y los requisitos de restricción, que actúan como restricciones que se imponen a la solución. A continuación se muestra los distintos campos que compondrán los requisitos y una breve descripción del mismo:

- **Identificador:** Nombra el requisito de manera única. Está formado por "RF" en el caso de los requisitos funcionales y por "RR" para los requisitos de restricción seguido de un guión y una numeración de tres dígitos.
- **Prioridad:** Indica la prioridad del requisito, indispensable para la planificación de la implementación del sistema por el desarrollador.
- **Fuente:** Indicara la fuente de la cual se a obtenido el requisito.
- **Necesidad:** Lo indispensable que es el requisito para el proyecto.
- **Claridad:** Si la funcionalidad del requisito está clara.
- **Verificabilidad:** El grado en el que se puede comprobar la aplicación de un requisito en el proyecto.
- **Estabilidad:** El grado de probabilidad de que el requisito pueda sufrir cambios a lo largo del proyecto.
- **Descripción:** Describe de forma clara y concisa la funcionalidad del requisito.

A continuación se procede a detallar los distintos requisitos de usuario. Para la especificación de los mismos, se hará empleo de unas tablas al fin de hacer más sencilla la interpretación de la información de estos. Dichas tablas contarán con los campos expuestos anteriormente. Se empezara detallando los requisitos funcionales y se dejará para el final los requisitos de restricción.

- Requisitos Funcionales:

Identificador: RF-001	
Prioridad: ☑ Alta ☐ Media ☐ Baja	**Fuente:** ☑ Jorge Blasco ☐ Carlos Fdez.
Necesidad: ☑ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☑ Alta ☐ Media ☐ Baja	**Verificabilidad:** ☑ Alta ☐ Media ☐ Baja
Estabilidad:	No sufrirá cambios a lo largo del proyecto.
Descripción:	La aplicación deberá mediar entre las comunicaciones HTTP que realice un cliente, reenviando las mismas al servidor Web al que estuviesen dirigidas.

Tabla 5 - RF-001

Identificador: RF-002	
Prioridad: ☑ Alta ☐ Media ☐ Baja	**Fuente:** ☑ Jorge Blasco ☐ Carlos Fdez.
Necesidad: ☑ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☑ Alta ☐ Media ☐ Baja	**Verificabilidad:** ☑ Alta ☐ Media ☐ Baja
Estabilidad:	No sufrirá cambios a lo largo del proyecto.
Descripción:	La aplicación deberá mediar entre las comunicaciones HTTP que realice un servidor Web, reenviando estas respuestas del servidor al cliente que realizó la petición de dicho recurso.

Tabla 6 - RF-002

Identificador: RF-003	
Prioridad: ☑ Alta ☐ Media ☐ Baja	**Fuente:** ☑ Jorge Blasco ☐ Carlos Fdez.
Necesidad: ☑ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☑ Alta ☐ Media ☐ Baja	**Verificabilidad:** ☑ Alta ☐ Media ☐ Baja
Estabilidad:	No sufrirá cambios a lo largo del proyecto.
Descripción:	Se deberá permitir que la aplicación sea capaz de atender a varios clientes de manera simultánea.

Tabla 7 - RF-003

Identificador: RF-004	
Prioridad: ☑ Alta ☐ Media ☐ Baja	**Fuente:** ☑ Jorge Blasco ☐ Carlos Fdez.
Necesidad: ☑ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☑ Alta ☐ Media ☐ Baja	**Verificabilidad:** ☑ Alta ☐ Media ☐ Baja
Estabilidad:	No sufrirá cambios a lo largo del proyecto.
Descripción:	La aplicación permitirá elegir, antes de su arranque, activar o desactivar tanto los filtros esteganográficos encargados de filtrar las peticiones y las respuestas HTTP.

Tabla 8 - RF-004

Identificador: RF-005	
Prioridad: ☑ Alta ☐ Media ☐ Baja	**Fuente:** ☑ Jorge Blasco ☐ Carlos Fdez.
Necesidad: ☑ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☑ Alta ☐ Media ☐ Baja	**Verificabilidad:** ☑ Alta ☐ Media ☐ Baja
Estabilidad:	No sufrirá cambios a lo largo del proyecto.
Descripción:	Se permitirá indicar el número del puerto por el cual se desea que se atiendan las peticiones de los clientes.

Tabla 9 - RF-005

Identificador: RF-006	
Prioridad: ☑ Alta ☐ Media ☐ Baja	**Fuente:** ☑ Jorge Blasco ☐ Carlos Fdez.
Necesidad: ☑ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☑ Alta ☐ Media ☐ Baja	**Verificabilidad:** ☑ Alta ☐ Media ☐ Baja
Estabilidad:	No sufrirá cambios a lo largo del proyecto.
Descripción:	La aplicación mostrará mensajes para la depuración del código de la aplicación. Dichos mensajes tendrán distintos niveles, tales como mensajes de información, de error, advertencias, etc. El administrador podrá elegir el nivel de los mensajes que desea ver.

Tabla 10 - RF-006

Identificador: RF-007	
Prioridad: ☑ Alta ☐ Media ☐ Baja	**Fuente:** ☑ Jorge Blasco ☐ Carlos Fdez.
Necesidad: ☑ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☑ Alta ☐ Media ☐ Baja	**Verificabilidad:** ☑ Alta ☐ Media ☐ Baja
Estabilidad:	No sufrirá cambios a lo largo del proyecto.
Descripción:	La aplicación filtrará posible información oculta en los archivos enviados mediante filtros esteganográficos. Estos filtros se crearán para el filtrado de imágenes y audio.

Tabla 11 - RF-007

Identificador: RF-008	
Prioridad: ☑ Alta ☐ Media ☐ Baja	**Fuente:** ☐ Jorge Blasco ☑ Carlos Fdez.
Necesidad: ☑ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☑ Alta ☐ Media ☐ Baja	**Verificabilidad:** ☑ Alta ☐ Media ☐ Baja
Estabilidad:	No sufrirá cambios a lo largo del proyecto.
Descripción:	Se creará un filtro capaz de eliminar información oculta en archivos de imagen GIF, habiendo sido embebida dicha información mediante la técnica gifshuffle.

Tabla 12 - RF-008

Identificador: RF-009	
Prioridad: ☑ Alta ☐ Media ☐ Baja	**Fuente:** ☐ Jorge Blasco ☑ Carlos Fdez.
Necesidad: ☑ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☑ Alta ☐ Media ☐ Baja	**Verificabilidad:** ☑ Alta ☐ Media ☐ Baja
Estabilidad:	No sufrirá cambios a lo largo del proyecto.
Descripción:	Se creará un filtro capaz de eliminar información oculta en archivos de imagen GIF, habiendo sido embebida dicha información mediante la técnica LSB.

Tabla 13 - RF-009

Identificador: RF-010	
Prioridad: ☑ Alta ☐ Media ☐ Baja	**Fuente:** ☐ Jorge Blasco ☑ Carlos Fdez.
Necesidad: ☑ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☑ Alta ☐ Media ☐ Baja	**Verificabilidad:** ☑ Alta ☐ Media ☐ Baja
Estabilidad:	No sufrirá cambios a lo largo del proyecto.
Descripción:	Se creará un filtro capaz de eliminar información oculta en archivos de imagen JPEG, habiendo sido embebida dicha información mediante la técnica LSB.

Tabla 14 - RF-010

Identificador: RF-011	
Prioridad: ☑ Alta ☐ Media ☐ Baja	**Fuente:** ☐ Jorge Blasco ☑ Carlos Fdez.
Necesidad: ☑ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☑ Alta ☐ Media ☐ Baja	**Verificabilidad:** ☑ Alta ☐ Media ☐ Baja
Estabilidad:	No sufrirá cambios a lo largo del proyecto.
Descripción:	Se creará un filtro capaz de eliminar información oculta en archivos de imagen BMP, habiendo sido embebida dicha información mediante la técnica LSB.

Tabla 15 - RF-011

Identificador: RF-012	
Prioridad: ☑ Alta ☐ Media ☐ Baja	**Fuente:** ☑ Jorge Blasco ☐ Carlos Fdez.
Necesidad: ☑ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☑ Alta ☐ Media ☐ Baja	**Verificabilidad:** ☑ Alta ☐ Media ☐ Baja
Estabilidad:	No sufrirá cambios a lo largo del proyecto.
Descripción:	Se creará un filtro capaz de eliminar información oculta en archivos de audio MP3, habiendo sido embebida dicha información mediante el empleo del programa MP3Stego (7).

Tabla 16 - RF-012

Identificador: RF-013	
Prioridad: ☑ Alta ☐ Media ☐ Baja	**Fuente:** ☑ Jorge Blasco ☐ Carlos Fdez.
Necesidad: ☑ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☑ Alta ☐ Media ☐ Baja	**Verificabilidad:** ☐ Alta ☑ Media ☐ Baja
Estabilidad:	No sufrirá cambios a lo largo del proyecto.
Descripción:	Los archivos filtrados por los filtros esteganográficos no deberán sufrir ningún cambio significativo, ni visual ni auditivo. Es decir, no se deberán notar cambios en las imágenes que delaten que han sido modificadas a simple vista, o cambios audibles en el caso de los ficheros de audio.

Tabla 17 - RF-013

Identificador: RF-014	
Prioridad: ☑ Alta ☐ Media ☐ Baja	**Fuente:** ☑ Jorge Blasco ☐ Carlos Fdez.
Necesidad: ☑ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☑ Alta ☐ Media ☐ Baja	**Verificabilidad:** ☑ Alta ☐ Media ☐ Baja
Estabilidad:	No sufrirá cambios a lo largo del proyecto.
Descripción:	Se debe mostrar un error y salir de la aplicación cuando los parámetros introducidos en el fichero de configuración no sean válidos. Mostrará también un ejemplo correcto de parámetros válidos para el fichero de configuración

Tabla 18 - RF-014

Identificador: RF-015	
Prioridad: ☑ Alta ☐ Media ☐ Baja	**Fuente:** ☐Jorge Blasco ☑ Carlos Fdez.
Necesidad: ☐Esencial ☑ Deseable ☐ Opcional	
Claridad: ☑ Alta ☐ Media ☐ Baja	**Verificabilidad:** ☑ Alta ☐ Media ☐ Baja
Estabilidad:	No sufrirá cambios a lo largo del proyecto.
Descripción:	La aplicación estará parametrizada. Leerá de un fichero el número del puerto por el cual atenderá las peticiones, la activación o no activación de los filtros de entrada y de salida, y el nivel de actuación de dichos filtros. También leerá el nivel de depuración del código.

Tabla 19 - RF-015

- Requisitos Funcionales:

Identificador: RR-001	
Prioridad: ☑ Alta ☐ Media ☐ Baja	**Fuente:** ☑ Jorge Blasco ☐ Carlos Fdez.
Necesidad: ☑ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☑ Alta ☐ Media ☐ Baja	**Verificabilidad:** ☑ Alta ☐ Media ☐ Baja
Estabilidad:	No sufrirá cambios a lo largo del proyecto.
Descripción:	La versión del protocolo HTTP que usará la aplicación será la 1.1

Tabla 20 - RR-001

Identificador: RR-002	
Prioridad: ☑ Alta ☐ Media ☐ Baja	**Fuente:** ☑ Jorge Blasco ☐ Carlos Fdez.
Necesidad: ☑ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☑ Alta ☐ Media ☐ Baja	**Verificabilidad:** ☑ Alta ☐ Media ☐ Baja
Estabilidad:	No sufrirá cambios a lo largo del proyecto.
Descripción:	El fichero de configuración de la aplicación se llamará “Configure”.

Tabla 21 - RR-002

Identificador: RR-003	
Prioridad: ☑ Alta ☐ Media ☐ Baja	**Fuente:** ☑ Jorge Blasco ☐ Carlos Fdez.
Necesidad: ☑ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☑ Alta ☐ Media ☐ Baja	**Verificabilidad:** ☑ Alta ☐ Media ☐ Baja
Estabilidad:	No sufrirá cambios a lo largo del proyecto.
Descripción:	El fichero con los parámetros estará formado por una línea para cada parámetro. Cada línea se compondrá de un encabezado que represente el parámetro, seguido de dos puntos y el valor de dicho parámetro.

Tabla 22 - RR-003

Identificador: RR-004	
Prioridad: ☑ Alta ☐ Media ☐ Baja	**Fuente:** ☑ Jorge Blasco ☐ Carlos Fdez.
Necesidad: ☑ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☑ Alta ☐ Media ☐ Baja	**Verificabilidad:** ☑ Alta ☐ Media ☐ Baja
Estabilidad:	No sufrirá cambios a lo largo del proyecto.
Descripción:	El parámetro puerto del fichero de configuración deberá tener siempre un valor mayor que cero.

Tabla 23 - RR-004

Identificador: RR-005	
Prioridad: ☑ Alta ☐ Media ☐ Baja	**Fuente:** ☑ Jorge Blasco ☐ Carlos Fdez.
Necesidad: ☑ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☑ Alta ☐ Media ☐ Baja	**Verificabilidad:** ☑ Alta ☐ Media ☐ Baja
Estabilidad:	No sufrirá cambios a lo largo del proyecto.
Descripción:	Los parámetros de activación de los depuradores, tanto en las solicitudes como en las respuestas, tendrán como valor “SI” sin están activados y “NO” en caso contrario.

Tabla 24 - RR-005

Identificador: RR-006	
Prioridad: ☑ Alta ☐ Media ☐ Baja	**Fuente:** ☑ Jorge Blasco ☐ Carlos Fdez.
Necesidad: ☑ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☑ Alta ☐ Media ☐ Baja	**Verificabilidad:** ☑ Alta ☐ Media ☐ Baja
Estabilidad:	No sufrirá cambios a lo largo del proyecto.
Descripción:	Los parámetros sobre el nivel de actuación de los filtros, tanto en las solicitudes como en las respuestas, tendrán un valor comprendido entre 0 y 8, siendo ocho el valor más potente de actuación.

Tabla 25 - RR-006

Identificador: RR-007	
Prioridad: ☑ Alta ☐ Media ☐ Baja	**Fuente:** ☐ Jorge Blasco ☑ Carlos Fdez.
Necesidad: ☑ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☑ Alta ☐ Media ☐ Baja	**Verificabilidad:** ☑ Alta ☐ Media ☐ Baja
Estabilidad:	No sufrirá cambios a lo largo del proyecto.
Descripción:	El uso del proxy esteganográfico no deberá hacer que las conexiones se vean afectadas de una fuerte pérdida de rendimiento, aunque una pérdida moderada es justificable.

Tabla 26 - RR-007

Identificador: RR-008	
Prioridad: ☑ Alta ☐ Media ☐ Baja	**Fuente:** ☐ Jorge Blasco ☑ Carlos Fdez.
Necesidad: ☑ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☑ Alta ☐ Media ☐ Baja	**Verificabilidad:** ☑ Alta ☐ Media ☐ Baja
Estabilidad:	No sufrirá cambios a lo largo del proyecto.
Descripción:	El parámetro “Nivel logger” del fichero de configuración tendrá como valores: OFF, SEVERE, WARNING, INFO, ALL, FINEST, FINER, FINE y CONFIG.

Tabla 27 - RR-008

2.2.2 Subsistemas

La aplicación está compuesta por dos subsistemas independientes que interactúan entre sí. El primero de esos subsistemas es el proxy HTTP. Este subsistema es el encargado de la intercepción de todas las comunicaciones que se produzcan en la red mediante el protocolo HTTP. Por otro lado se cuenta con el subsistema de los filtros esteganográficos, conformado por los distintos filtros que se aplican a los datos enviados mediante HTTP. Ambos subsistemas se relacionan entre sí, ya que el proxy HTTP utiliza los filtros esteganográficos para filtrar los datos que recibe y posteriormente retransmite.

Los dos subsistemas están planteados para poder ser utilizados independientemente, o ser expandidos y actualizados en un futuro.

2.3 Plan de Pruebas

En esta sección se expondrá el plan de prueba. Dicho plan servirá para comprobar que la aplicación cumple con todos los requisitos formulados en la sección "2.2.1 Especificación de Requisitos de Usuario", así como para validar el sistema antes de su implantación y puesta en funcionamiento.

2.3.1 Definición de Pruebas de Validación

En este apartado se describen las pruebas que deberá pasar la aplicación, tras la fase de implementación, para comprobar el correcto funcionamiento de la misma, así como el cumplimiento de todos los requisitos postulados. Para la definición de las pruebas se va a utilizar tablas que contendrán el código identificativo de la prueba, formado por la palabra "PRU" seguida de un guión y una numeración formada por tres dígitos. También contendrá un campo que describa detalladamente la prueba a realizar, y un último campo con los requisitos funcionales que valida la prueba.

A continuación se pasa a ofrecer dichas pruebas:

Identificador: **PRU-001**	
Descripción:	Acceder mediante el empleo de un navegador Web a una página Web y comprobar su correcto cargado en el navegador.
Requisitos Relacionados:	RF-001, RF-002

Tabla 28 - PRU-001

Identificador: **PRU-002**	
Descripción:	Rellenar un formulario, o subir algún archivo al correo electrónico de Hotmail (Microsoft®), o realizar cualquier acción con el navegador que emplee el comando Post de HTTP.
Requisitos Relacionados:	RF-001, RF-002

Tabla 29 - PRU-002

Identificador: **PRU-003**	
Descripción:	Acceder a varias páginas web de manera simultánea y comprobar que todas se cargan a la vez correctamente.
Requisitos Relacionados:	RF-003

Tabla 30 - PRU-003

Identificador: **PRU-004**	
Descripción:	Acceder a una página web con el filtrado de respuestas activado.
Requisitos Relacionados:	RF-004

Tabla 31 - PRU-004

Identificador: **PRU-005**	
Descripción:	Acceder a una página web con el filtrado de respuestas desactivado.
Requisitos Relacionados:	RF-004

Tabla 32 - PRU-005

Identificador: **PRU-006**	
Descripción:	Rellenar un formulario, o subir algún archivo al correo electrónico de Hotmail (Microsoft®), o realizar cualquier acción con el navegador que emplee el comando Post de HTTP, con el filtrado de solicitudes activado y comprobar el correcto funcionamiento del mismo.
Requisitos Relacionados:	RF-004

Tabla 33 - PRU-006

Identificador: **PRU-007**	
Descripción:	Rellenar un formulario, o subir algún archivo al correo electrónico de Hotmail (Microsoft®), o realizar cualquier acción con el navegador que emplee el comando Post de HTTP, con el filtrado de solicitudes desactivado y comprobar el correcto funcionamiento del mismo.
Requisitos Relacionados:	RF-004

Tabla 34 - PRU-007

Identificador: **PRU-008**	
Descripción:	Modificar el puerto del proxy a un puerto distinto (por ejemplo el puerto 8080) e intentar acceder a una página Web mediante el uso del proxy a través de dicho puerto.
Requisitos Relacionados:	RF-005

Tabla 35 - PRU-008

Identificador: **PRU-009**	
Descripción:	Probar distintos niveles de depuración del código de la aplicación y comprobar que los mensajes mostrados son los deseados.
Requisitos Relacionados:	RF-006

Tabla 36 - PRU-009

Identificador: **PRU-010**	
Descripción:	Acceder a una página Web sin usar el proxy y guardar una imagen de la misma. Acceder usando el proxy y los filtros y guardar la misma imagen. Comprobar que ambas imágenes no son exactamente iguales.
Requisitos Relacionados:	RF-007

Tabla 37 - PRU-010

Identificador: **PRU-011**	
Descripción:	Usar el protocolo HTTP para enviar una imagen GIF esteganografiada con el programa Gifshuffle (3), y comprobar que se elimina la información del mismo tras pasar por el proxy.
Requisitos Relacionados:	RF-008

Tabla 38 - PRU-011

Identificador: **PRU-012**	
Descripción:	Usar el protocolo HTTP para enviar una imagen GIF esteganografiada mediante el método LSB, y comprobar que se elimina la información del mismo tras pasar por el proxy.
Requisitos Relacionados:	RF-009

Tabla 39 - PRU-012

Identificador: **PRU-013**	
Descripción:	Usar el protocolo HTTP para enviar una imagen JPEG esteganografiada mediante la aplicación JPHIDE (6), y comprobar que se elimina la información del mismo tras pasar por el proxy.
Requisitos Relacionados:	RF-010

Tabla 40 - PRU-013

Identificador: **PRU-014**	
Descripción:	Usar el protocolo HTTP para enviar una imagen BMP esteganografiada mediante la aplicación BlindSide (5), y comprobar que se elimina la información del mismo tras pasar por el proxy.
Requisitos Relacionados:	RF-011

Tabla 41 - PRU-014

Identificador: **PRU-015**	
Descripción:	Usar el protocolo HTTP para enviar un archivo de audio esteganografiado mediante la aplicación MP3Stego (7), y comprobar que se elimina la información del mismo tras pasar por el proxy.
Requisitos Relacionados:	RF-0012

Tabla 42 - PRU-015

Identificador: **PRU-016**	
Descripción:	Usar el protocolo HTTP para enviar una imagen a través del proxy, con nieles de filtrado 1 y 2. Comprobar que apenas haya cambios visuales en la imagen.
Requisitos Relacionados:	RF-0013

Tabla 43 - PRU-016

Identificador: **PRU-017**	
Descripción:	Usar el protocolo HTTP para enviar un archivo de audio a través del proxy, con nieles de filtrado 1 y 2. Comprobar que apenas haya cambios auditivos en el sonido del archivo de audio.
Requisitos Relacionados:	RF-0013

Tabla 44 - PRU-017

Identificador: **PRU-018**	
Descripción:	Introducir parámetros incorrectos en el archivo de configuración del proxy y comprobar que la aplicación muestra un error indicando que los parámetros no son los correctos y mostrando un ejemplo de parámetros correctos.
Requisitos Relacionados:	RF-0014

Tabla 45 - PRU-018

Identificador: **PRU-019**	
Descripción:	Arrancar el proxy modificado los valores de los parámetros del fichero "Configure" y comprobar correcto funcionamiento de la aplicación.
Requisitos Relacionados:	RF-0015

Tabla 46 - PRU-019

2.3.2Matriz de Trazabilidad

En este apartado se mostrará una matriz de trazabilidad ente los requisitos de usuario definidos y las pruebas anteriores. Con la matriz se puede comprobar que todos los requisitos tienen como mínimo una prueba que los evalúe. De esta forma, se puede saber que el sistema funciona completamente si se pasan todas las pruebas.

	PRU-001	PRU-002	PRU-003	PRU-004	PRU-005	PRU-006	PRU-007	PRU-008	PRU-009	PRU-010	PRU-011	PRU-012	PRU-013	PRU-014	PRU-015	PRU-016	PRU-017	PRU-018	PRU-019
RF-001	X	X																	
RF-002	X	X																	
RF-003			X																
RF-004				X	X	X	X												
RF-005								X											
RF-006									X										
RF-007										X									
RF-008											X								
RF-009												X							
RF-010													X						
RF-011														X					
RF-012															X				
RF-013																X	X		
RF-014																		X	
RF-015																			X

Tabla 47 - M. Trazabilidad entre Requisitos y Pruebas

2.3.3Pruebas de Rendimiento

Con el fin de comprobar el rendimiento de la aplicación una vez finalizada, se deberán llevar a cabo una serie de pruebas de rendimiento, que certifiquen que la aplicación trabaja con un rendimiento aceptable, es decir, que la diferencia de tiempos entre una tarea sin y con el proxy esteganográfico no es muy elevada. A continuación se muestran las pruebas de rendimiento que deberán realizarse a la aplicación una vez finalizada. Dichos resultados deberán ser analizados para obtener conclusiones sobre el rendimiento de la aplicación.

Identificador: **PR-001**	
Descripción:	Acceder a www.google.es sin el proxy esteganográfico, con proxy y sin filtros, con proxy y con filtro de solicitudes, con proxy y con filtro de respuestas, con proxy y ambos filtros.

Tabla 48 - PR-001

Identificador: **PR-002**	
Descripción:	Acceder a www.hotmail.es sin el proxy esteganográfico, con proxy y sin filtros, con proxy y con filtro de solicitudes, con proxy y con filtro de respuestas, con proxy y ambos filtros.

Tabla 49 - PR-002

Identificador: **PR-003**	
Descripción:	Acceder a www.elpais.es sin el proxy esteganográfico, con proxy y sin filtros, con proxy y con filtro de solicitudes, con proxy y con filtro de respuestas, con proxy y ambos filtros.

Tabla 50 - PR-003

Identificador: **PR-004**	
Descripción:	Usando una cuenta de correo en www.hotmail.com, enviar una imagen (GIF,JPEG y BMP), y un archivo de audio (MP3) por correo sin el proxy esteganográfico, con proxy y sin filtros, con proxy y con filtro de solicitudes, con proxy y con filtro de respuestas, con proxy y ambos filtros.

Tabla 51 - PR-004

Capítulo 3
Diseño del Proyecto

Este capítulo define con detalle las funcionalidades de la aplicación, mostrando la solución al problema planteado en la fase de análisis, explicando las técnicas y algoritmos concretos que se utilizarán. Es durante este proceso en el que se detalla la arquitectura del sistema de manera exhaustiva, especificando detalladamente los distintos componentes que conforman el sistema, que serán explicados de forma detallada, indicando toda la información necesaria para la definición de los mismos, que será utilizada en la implementación de la aplicación.

Además, en esté capitulo también se explicarán las herramientas que se utilizarán para el desarrollo del software.

3.1 Arquitectura del Proyecto

La arquitectura utilizada en el proyecto es la arquitectura cliente-servidor con una gran peculiaridad, que es que la aplicación realiza tanto la labor de cliente como la de servidor. Como la aplicación que desea desarrollar este proyecto se trata de un proxy Web, ésta debe mediar entre las comunicaciones que realice un cliente Web con un servidor. Por esa razón, el proxy debe comportarse como un servidor para el cliente Web, y como un cliente Web para con el servidor.

En su papel como servidor Web, el proxy recibe una petición HTTP por parte de un cliente, en el se le solicita un recurso Web. Inmediatamente, el proxy pasa a comportarse como un cliente, solicitando al servidor Web el recurso que necesita para contestar al cliente. Una vez recibido dicho recurso, vuelve a comportarse como un servidor, remitiendo al cliente Web el recurso que había solicitado.

El rol de intermediario entre clientes y servidores web que desarrolla el proxy, nos permite filtrar las comunicaciones HTTP que se produzcan, permitiendo así la consecución del principal objetivo de este proyecto, eliminar de los datos enviados mediante el protocolo HTTP cualquier información embebida en los mismos mediante técnicas esteganográficas.

Los filtros esteganográficos actuarán en dos momentos de la comunicación entre el cliente y el servidor. Primero cuando el proxy ha recibido la petición de un cliente, y antes de ser remitida al servidor, y segundo cuando recibe la respuesta del servidor, antes de enviársela al cliente. A continuación se muestra una ilustración del funcionamiento de la aplicación.

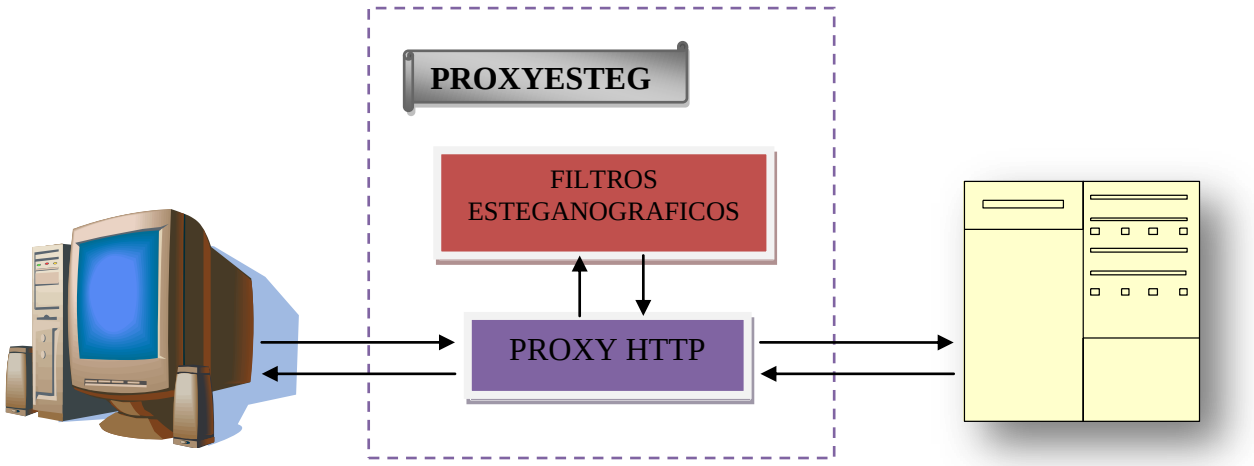

Ilustración 14 - Arquitectura del Proyecto

3.2 Herramientas de Desarrollo del Software

En esta sección enumerarán las herramientas software empleadas para la realización de este proyecto:

- **Netbeans IDE 6.5:** Plataforma, de código abierto, para el desarrollo de aplicaciones.
- **Microsoft Office Word 2007 ®:** Procesador de texto de la suite ofimática de Microsoft ®.
- **Microsoft Office Visio 2007 ®:** Editor de diagramas de la suite ofimática de Microsoft ®.
- **Microsoft Project 2007®:** Software de administración de proyectos.

La aplicación Netbeans ha sido usada para la implementación de la aplicación. Tanto la aplicación Microsoft Office Word 2007, como Microsoft Project 2007, como Microsoft Office Visio 2007 han sido usadas para la creación de la documentación del proyecto.

3.3 Modelado de la Arquitectura Estática

El modelado de la arquitectura estática del sistema, muestra los elementos que conforman la parte estática del sistema. Dichos elementos han sido obtenidos de los requisitos de la fase de análisis. La arquitectura estática del sistema está compuesta por las clases de este, indicando los atributos y métodos de las mismas, y por el diagrama de clases formado por todas las clases del sistema.

3.3.1 Diagrama de Clases

En este apartado se muestra el diagrama de clases de la aplicación. Dicho diagrama sirve para mostrar las clases, atributos y métodos del sistema, así como la relación existente entre estos. Se empezara mostrando el diagrama de los paquetes de la aplicación, indicando las dependencias entre unos paquetes y otros, para acto seguido profundizar y pasar a mostrar los diagramas de clases de los distintos paquetes.

- Diagrama de paquetes:

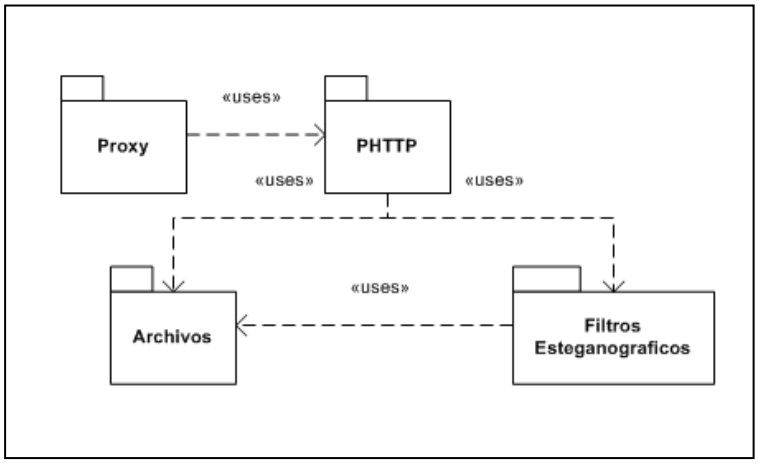

Ilustración 15 - Diagrama de paquetes

- Diagrama de la clase Proxy:

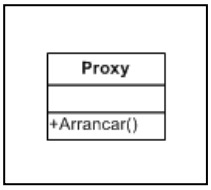

Ilustración 16 - Diagrama de clases (Proxy)

- Diagrama de la clase FiltrosEsteganograficos:

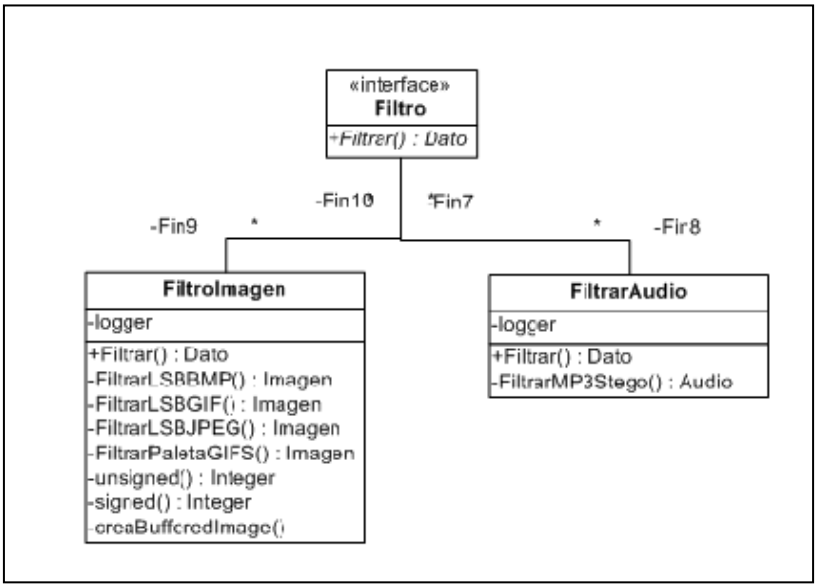

Ilustración 17 - Diagrama de clases (FiltrosEsteganograficos)

- Diagrama de la clase Archivos:

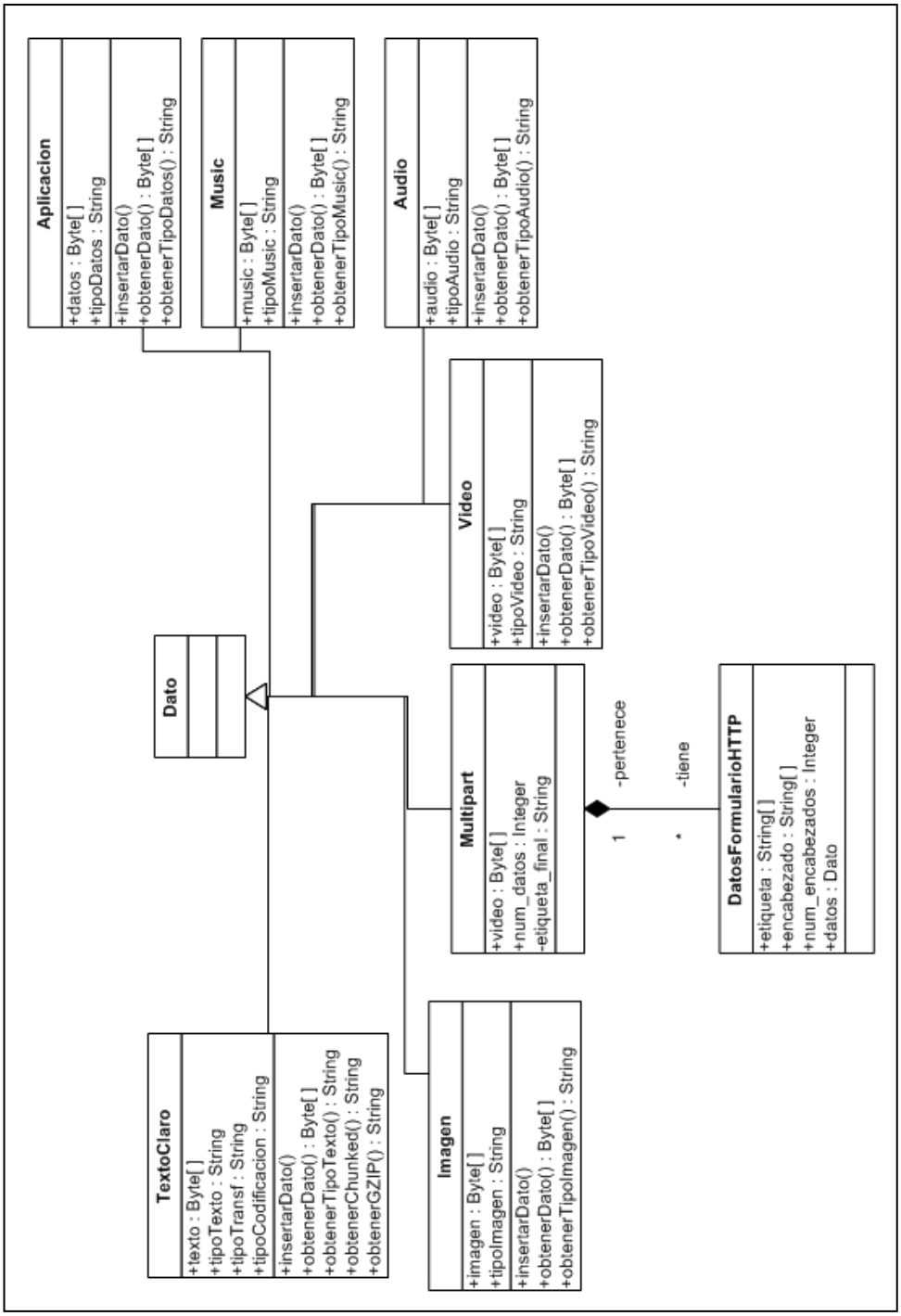

Ilustración 18 - Diagrama de clases (Archivos)

- Diagrama de la clase:

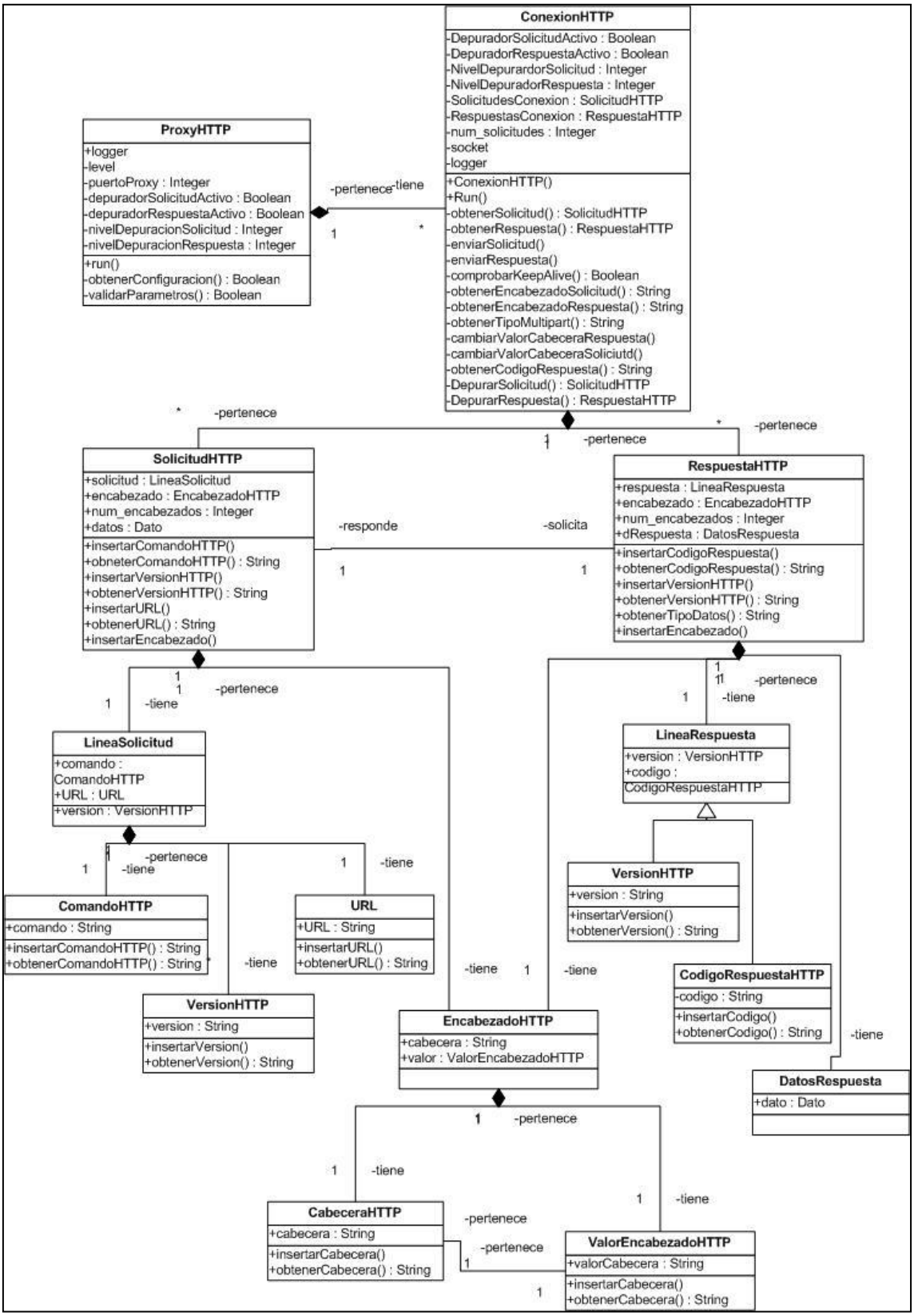

Ilustración 19 - Diagrama de clases (PHTTP)

3.3.2Clases, Atributos y Métodos

En este apartado se detallan las distintas clases que forman el sistema, junto a sus atributos y métodos. Para cada clase se utilizarán sendas tablas, una para los atributos y otra para los métodos, detallando en cada caso los mismos mediante una descripción de estos y otros datos de interés. A continuación se muestran las clases que componen el proyecto.

- **Proxy:**

La clase Proxy arranca el proxy HTTP de la aplicación. Esta clase no cuenta con atributos, aunque sí que cuenta con un único método, encargado de arrancar el proxy HTTP. Esta clase está englobada en el paquete Proxy.

Métodos				
Nombre	**Privacidad**	**Retorno**	**Parámetros**	**Descripción**
Arrancar	Public	Void	Ninguno	Arranca el proxy HTTP creando un nuevo hilo para ello.

Tabla 52 - Métodos clase Proxy

- **ProxyHTTP:**

La clase ProxyHTTP es la encargada de recibir las peticiones http formuladas por los clientes, así como de configurar el proxy con los parámetros establecidos por el usuario. Esta clase está englobada dentro del paquete PHTTP.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
depuradorSolicitudActivo	Private	boolean	Indica si el proxy debe tener activada la depuración de solicitudes.
depuradorRespuestaActivo	Private	Boolean	Indica si el proxy debe tener activada la depuración de respuestas.
level	Private	Level	Nivel de depuración usado en la clase Logger.
logger	Public	Logger	Depurador de código usado para mostrar mensajes por pantalla.
NivelDepuracionSolicitud	Private	Int	Nivel de actuación del depurador de solicitudes.
NivelDepuracionRespuesta	Private	Int	Nivel de actuación del depurador de respuestas.
puertoProxy	Private	Int	Número del puerto por el cual atiende el proxy las peticiones de los clientes.

Tabla 53 - Atributos Clase ProxyHTTP

Métodos				
Nombre	**Privacidad**	**Retorno**	**Parámetros**	**Descripción**
run	Public	Void	Ninguno	Configura el proxy HTTP y gestiona las solicitudes de los clientes creando hilos para atenderlas.
obtenerConfiguracion	Private	Boolean	Ninguno	Obtiene la configuración del fichero configure y la almacena para ser usada. Devuelve verdadero si todo fue correcto.
validarParametros	Private	Boolean	Error: boolean	Valida los parámetros y devuelve true si todos ellos son correctos.

Tabla 54 - Métodos clase ProxyHTTP

- **ConexionHTTP:**

La clase ConexionHTTP atiende las peticiones de los clientes, realizando todas las labores de comunicación tanto con el cliente como con el servidor. También invoca los filtros, tanto para las solicitudes como para las respuestas. Se encuentra en el paquete PHTTP.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
DepuradorSolicitudActivo	Private	boolean	Indica si el proxy debe tener activada la depuración de solicitudes.
DepuradorRespuestaActivo	Private	Boolean	Indica si el proxy debe tener activada la depuración de respuestas.
logger	Public	Logger	Depurador de código usado para mostrar mensajes por pantalla.
NivelDepuracionSolicitud	Private	Int	Nivel de actuación del depurador de solicitudes.
NivelDepuracionRespuesta	Private	Int	Nivel de actuación del depurador de respuestas.
num_solicitudes	Private	Int	Número de solicitudes atendidas
SolicitudesConexion	Private	Solicitud HTTP[ ]	Datos de las solicitudes HTTP atendidas en una conexión.
Socket	Private	Socket	Socket por el que se recibe la solicitud.
RespuestaConexion	Private	Respuesta HTTP[ ]	Datos de las respuestas HTTP atendidas en una conexión.

Tabla 55 - Atributos clase ConexionHTTP

Métodos				
Nombre	**Privacidad**	**Retorno**	**Parámetros**	**Descripción**
ConexionHTTP	Public	Void	Socket: Socket DepuradorSolicitudActivo: boolean DepuradorRespuestaActivo: boolean dSolicitud: int dRespuesta: int	Constructor de la clase ConexiónHTTP.
run	Public	Void	Ninguno	Comienza la atención de una conexión HTTP.
obtenerSolicitud	Private	Solicitud HTTP	Buffer:DataInputStream	Obtiene una solicitud HTTP de un buffer de entrada.
comprobarKeepAlive	Private	Boolean	Ninguna	Devuelve si la conexión es persistente.
enviarSolicitud	Private	Void	solicitud: SolicitudHTTP socket: Socket	Envía una solicitud HTTP mediante el empleo de un socket.
obtenerRespuesta	Private	Respuesta HTTP	Buffer: DataInputStream	Obtiene una respuesta HTTP de un buffer de entrada.
enviarRespuesta	Private	Void	respuesta: RespuestaHTTP socket: Socket	Envía una respuesta HTTP mediante el empleo de un socket.
obtenerEncabezado Solicitud	Private	String	solicitud: SolicitudHTTP encabezado: String	Devuelve una cadena con el valor del encabezado deseado de la solicitud.
obtenerEncabezado Respuesta	Private	String	respuesta: respuesta: HTTP encabezado: String	Devuelve una cadena con el valor del encabezado deseado de la respuesta.
obtenerTipoMultipart	Private	String	Datos: DatosFormularioHTTP	Devuelve una cadena con el tipo de datos de un mensaje multipart.
cambiarValorCabecera Solicitud	Private	Void	Respuesta: SolicitudHTTP Cabecera: String Valor: Int	Cambia el valor de una cabecera por el deseado.
cambiarValorCabecera Respuesta	Private	Void	Respuesta: RespuestaHTTP Cabecera: String Valor: Int	Cambia el valor de una cabecera por el deseado.
obtenerCodigo Respuesta	Private	String	Respuesta: RespuestaHTTP	Devuelve el código de la respuesta.
DepurarSolicitud HTTP	Private	Solicitud HTTP	Solicitud: SolicitudHTTP	Depura una solicitud HTTP.
DepurarRespuesta HTTP	Private	Respuesta HTTP	Respuesta: RespuestaHTTP	Depura un respuesta HTTP.

Tabla 56 - Métodos clase ConexionHTTP

- **SolicitudHTTP:**

La clase SolicitudHTTP almacena las solicitudes realizadas por los clientes. Pertenece al paquete PHTTP.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
Solicitud	Public	Linea Solicitud	La línea principal de una solicitud.
Encabezado	Public	EncabezadoHTTP[]	Los encabezados que tiene una solicitud.
Num_encabezados	Public	Int	El número de encabezados que tiene una solicitud.
datos	Public	Dato	Los datos de una solicitud.

Tabla 57 - Atributos clase SolicitudHTTP

Métodos				
Nombre	**Privacidad**	**Retorno**	**Parámetros**	**Descripción**
InsertarComandoHTTP	Public	Void	Comando: String	Inserta un comando.
obtenerComandoHTTP	Public	String	Ninguno	Obtiene un comando.
insertarVersionHTTP	Public	Void	Versión: String	Inserta la versión HTTP.
obtenerVersionHTTP	Public	String	Ninguno	Obtiene la versión HTTP.
insertarURL	Public	Void	URL: String	Inserta una URL.
obtenerURL	Public	String	Ninguno	Obtiene una URL.
insertarEncabezado	Public	Void	Cabecera: String Valor: String	Inserta un encabezado HTTP junto a su valor.

Tabla 58 - Métodos clase SolicitudHTTP

- **LineaSolicitud:**

La clase LineaSolicitud contiene los datos de las líneas de solicitud HTTP, siendo estas, el comando, la URL, y la versión. Pertenece al paquete PHTTP.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
Comando	Public	Comando HTTP	Contiene el comando HTTP de la solicitud.
URL	Public	URL	Contiene la URL de la solicitud.
Version	Public	Version HTTP	Contiene la versión HTTP de la solicitud.

Tabla 59 - Atributos clase LíneaSolicitud

- **ComandoHTTP:**

La clase ComandoHTTP almacena el comando de una petición HTTP. Se incluye en el paquete PHTTP.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
Comando	Public	String	Contiene el comando HTTP de la solicitud.

Tabla 60 - Atributos clase ComandoHTTP

Métodos				
Nombre	**Privacidad**	**Retorno**	**Parámetros**	**Descripción**
InsertarComando HTTP	Public	Void	Comando: String	Inserta un comando.
obtenerComando HTTP	Public	String	Ninguno	Obtiene un comando.

Tabla 61 - Métodos clase ComandoHTTP

- **URL:**

La clase URL almacena la URL de una petición HTTP. Se incluye en el paquete PHTTP.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
URL	Public	String	Contiene la URL de la solicitud.

Tabla 62 - Atributos clase URL

Métodos				
Nombre	**Privacidad**	**Retorno**	**Parámetros**	**Descripción**
insertarURL	Public	Void	URL: String	Inserta una URL.
obtenerURL	Public	String	Ninguno	Obtiene una URL.

Tabla 63 - Métodos clase URL

- **VersionHTTP:**

La clase VersionHTTP almacena la versión de una petición HTTP. Se incluye en el paquete PHTTP.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
Versión	Public	String	Contiene la versión HTTP de la solicitud.

Tabla 64 - Atributos clase VersionHTTP

Métodos				
Nombre	**Privacidad**	**Retorno**	**Parámetros**	**Descripción**
insertarVersion	Public	Void	URL: String	Inserta una URL.
obtenerVersion	Public	String	Ninguno	Obtiene una URL.

Tabla 65 - Métodos clase VersionHTTP

- **EncabezadoHTTP:**

La clase EncabezadoHTTP almacena cabeceras, tanto el encabezado como el valor de estas. Forma parte del paquete PHTTP.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
cabecera	Public	Cabecera HTTP	El encabezado de una cabecera.
valor	Public	Valor EncabezadoHTTP	El valor de un encabezado HTTP.

Tabla 66 - Atributos clase EncabezadoHTTP

- **CabeceraHTTP:**

La clase CabeceraHTTP almacena el encabezado de una cabecera. Forma parte del paquete PHTTP.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
Cabecera	Public	String	El encabezado de una cabecera.

Tabla 67 - Atributos clase CabeceraHTTP

Métodos				
Nombre	**Privacidad**	**Retorno**	**Parámetros**	**Descripción**
insertarCabecera	Public	Void	cabecera: String	Inserta un encabezado.
obtenerCabecera	Public	String	Ninguno	Obtiene un encabezado.

Tabla 68 - Métodos clase CabeceraHTTP

- **ValorEncabezadoHTTP:**

La clase ValorEncabezadoHTTP almacena el valor de un encabezado. Forma parte del paquete PHTTP.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
valorCabecera	Public	String	El valor de un encabezado HTTP.

Tabla 69 - Atributos clase ValorEncabezadoHTTP

Métodos				
Nombre	**Privacidad**	**Retorno**	**Parámetros**	**Descripción**
ValorEncabezadoHTTP	Public		valor: String	Constructor de la clase.
insertarValorCabecera	Public	Void	valor: String	Inserta el valor de un encabezado.
obtenerValorCabecera	Public	String	Ninguno	Obtiene el valor de un encabezado.

Tabla 70 - Métodos clase ValorEncabezadoHTTP

- **RespuestaHTTP:**

La clase RespuestaHTTP almacena una respuesta HTTP. Forma parte del paquete PHTTP.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
respuesta	Public	Linea Respuesta	La línea principal de una respuesta.
Encabezado	Public	EncabezadoHTTP[]	Los encabezados que tiene una respuesta.
Num_encabezados	Public	Int	El número de encabezados que tiene una respuesta.
dRespuesta	Public	Datos Respuesta	Los datos de una respuesta.

Tabla 71 - Atributos clase RespuestaHTTP

Métodos				
Nombre	**Privacidad**	**Retorno**	**Parámetros**	**Descripción**
InsertarCodigo Respuesta	Public	Void	Comando: String	Inserta el código de una respuesta.
obtenerCodigo Respuesta	Public	String	Ninguno	Obtiene el código de una respuesta.
insertarVersionHTTP	Public	Void	Versión: String	Inserta la versión HTTP.
obtenerVersionHTTP	Public	String	Ninguno	Obtiene la versión HTTP.
ObtenerTipoDatos	Public	String	Ninguno	Obtiene el tipo de los datos.
insertarEncabezado	Public	Void	Cabecera: String Valor: String	Inserta un encabezado HTTP junto a su valor.

Tabla 72 - Métodos clase RespuestaHTTP

- **LineaRespuesta:**

La clase LineaRespuesta contiene la línea de respuesta de una respuesta HTTP. Forma parte del paquete PHTTP.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
Version	Public	Version HTTP	Contiene la versión HTTP de la respuesta.
Código	Public	Codigo Respuesta	Contiene el código de respuesta.

Tabla 73 - Atributos clase LíneaRespuesta

- **CodigoRespuestaHTTP:**

La clase CodigoRespuestaHTTP almacena el valor de un código de respuesta. Forma parte del paquete PHTTP.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
codigoRespuesta	Public	String	Contiene el código de respuesta.

Tabla 74 - Atributos clase CodigoRespuestaHTTP

Métodos				
Nombre	**Privacidad**	**Retorno**	**Parámetros**	**Descripción**
insertarCodigo	Public	Void	codigo: String	Inserta el valor de un código de respuesta.
obtenerCodigo	Public	String	Ninguno	Obtiene el valor de un código de respuesta.

Tabla 75 - Métodos clase CodigorRespuestaHTTP

- **DatosRespuesta:**

La clase DatosRespuesta contiene los datos de una respuesta. Pertenece al paquete PHTTP.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
dato	Public	Datos	Contiene los datos de la respuesta.

Tabla 76 - Atributos clase DatosRespuesta

- **Imagen:**

La clase Imagen sirve para almacenar una imagen, incluyendo el tipo específico de esta. Pertenece al paquete Archivos.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
imagen	Public	Byte[ ]	Datos que conforman la imagen.
tipoImagen	Public	String	Tipo concreto de imagen.

Tabla 77 - Atributos clase Imagen

Métodos				
Nombre	**Privacidad**	**Retorno**	**Parámetros**	**Descripción**
insertarDato	Public	Void	codigo: byte[ ] tipoImagen: String	Inserta una imagen, incluyendo sus datos y tipo concreto.
obtenerDato	Public	Byte [ ]	Ninguno	Obtiene los datos de la imagen.
obtenerTipoImagen	Public	String	Ninguno	Obtiene el tipo concreto de la imagen.

Tabla 78 - Métodos clase Imagen

- **Video:**

La clase Video sirve para almacenar una video, incluyendo el tipo específico de este. Pertenece al paquete Archivos.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
video	Public	Byte[ ]	Datos que conforman el video.
tipoVideo	Public	String	Tipo concreto de video.

Tabla 79 - Atributos clase Video

Métodos				
Nombre	**Privacidad**	**Retorno**	**Parámetros**	**Descripción**
insertarDato	Public	Void	codigo: byte[ ] tipoVideo: String	Inserta una video, incluyendo sus datos y tipo concreto.
obtenerDato	Public	Byte [ ]	Ninguno	Obtiene los datos del video.
obtenerTipoVideo	Public	String	Ninguno	Obtiene el tipo concreto del video.

Tabla 80 - Métodos clase Video

- **Audio:**

La clase Audio sirve para almacenar una imagen, incluyendo el tipo específico de esta. Pertenece al paquete Archivos.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
audio	Public	Byte[ ]	Datos que conforman el audio.
tipoAudio	Public	String	Tipo concreto de audio.

Tabla 81 - Atributos clase Audio

Métodos				
Nombre	**Privacidad**	**Retorno**	**Parámetros**	**Descripción**
insertarDato	Public	Void	codigo: byte[ ] tipoAudio: String	Inserta un audio, incluyendo sus datos y tipo concreto.
obtenerDato	Public	Byte [ ]	Ninguno	Obtiene los datos del audio.
obtenerTipoAudio	Public	String	Ninguno	Obtiene el tipo concreto del audio.

Tabla 82 - Métodos clase Audio

- **TextoClaro:**

La clase TextoClaro sirve para almacenar texto, incluyendo el tipo específico, tipo de transferencia y el tipo de codificación de este. Pertenece al paquete Archivos.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
texto	Public	Byte[ ]	Datos que conforman el texto..
tipoTexto	Public	String	Tipo concreto de texto.
tipoTransf	Public	String	Tipo de transferencia del texto.
tipoCodificacion	Public	String	Tipo de codificación del texto.

Tabla 83 - Atríbutos clase TextoClaro

Métodos				
Nombre	**Privacidad**	**Retorno**	**Parámetros**	**Descripción**
insertarDato	Public	Void	texto: byte[ ] tipoTexto: String tipoTransf: String tipoCodificación: String	Inserta un texto, incluyendo sus datos, tipo concreto, tipo transferencia y tipo codificación.
obtenerDato	Public	Byte [ ]	Ninguno	Obtiene los datos del texto.
obtenerTipoTexto	Public	String	Ninguno	Obtiene el tipo concreto del texto.
obtenerChunked	Public	String	Ninguno	Obtiene el tipo de transferencia.
obtenerGZIP	Public	String	Ninguno	Obtiene el tipo de codificación.

Tabla 84 - Métodos clase TextoClaro

- **Music:**

La clase Music sirve para almacenar música, incluyendo el tipo específico de este. Pertenece al paquete Archivos.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
music	Public	Byte[ ]	Datos que conforman la música.
tipoMusic	Public	String	Tipo concreto de música.

Tabla 85 - Atríbutos clase Music

Métodos				
Nombre	**Privacidad**	**Retorno**	**Parámetros**	**Descripción**
insertarDato	Public	Void	music: byte[ ] tipoMusic: String	Inserta una música, incluyendo sus datos y tipo concreto.
obtenerDato	Public	Byte [ ]	Ninguno	Obtiene los datos de música.
obtenerTipoMusic	Public	String	Ninguno	Obtiene el tipo concreto de música.

Tabla 86 - Métodos clase Music

- **Aplicación:**

La clase Aplicación sirve para almacenar aplicaciones, incluyendo el tipo específico de esta. Pertenece al paquete Archivos.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
datos	Public	Byte[ ]	Datos que conforman la aplicación.
tipoDatos	Public	String	Tipo concreto de aplicación.

Tabla 87 - Atributos clase Aplicación

Métodos				
Nombre	**Privacidad**	**Retorno**	**Parámetros**	**Descripción**
insertarDato	Public	Void	datos: byte[ ] tipoDatos: String	Inserta una aplicación, incluyendo sus datos y tipo concreto.
obtenerDato	Public	Byte [ ]	Ninguno	Obtiene los datos de aplicación.
obtenerTipoDatos	Public	String	Ninguno	Obtiene el tipo concreto de aplicación.

Tabla 88 - Métodos clase Aplicación

- **Multipart:**

La clase Multipart sirve para almacenar información mixta, incluyendo el tipo específico de este. Pertenece al paquete Archivos.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
datos	Public	Datos Formulario HTTP[ ]	Almacena los distintos datos que forman el formulario.
Num_datos	Public	Int	Número de datos almacenados.
etiquetaFinal	Public	String	Etiqueta final de la comunicación.

Tabla 89 - Atributos clase Multipart

- **DatosFormularioHTTP:**

La clase DatosFormularioHTTP almacena los distintos datos que se envían en un formulario.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
etiqueta	Public	String	Etiqueta de inicio de sección de datos.
Encabezado	Public	String[ ]	Encabezados de la sección.
Num_encabezados	Public	String	Número de encabezados de la sección.
Datos	Public	Datos	Información de la sección.

Tabla 90 - Atributos clase DatosFormularioHTTP

- **Filtro:**

La clase Filtro es una interfaz del paquete FiltrosEsteganograficos, encargada de ofrecer soporte para las clases FiltroImagen y FiltroAudio.

Métodos				
Nombre	**Privacidad**	**Retorno**	**Parámetros**	**Descripción**
Filtrar	Public	Dato	dato: Dato NivelDepuracion: Int	Es la interfaz de otros métodos de otras clases.

Tabla 91 - Métodos interfaz Filtro

- **FiltroImagen:**

La clase FiltroImagen es utilizada para filtrar imágenes que tengan embebido otro mensaje mediante técnicas esteganográficas. Pertenece al paquete FiltrosEsteganográficos.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
Logger	Private	Logger	Se utiliza para mostrar mensajes de depuración por pantalla.

Tabla 92 - Atributos clase FiltroImagen

Métodos				
Nombre	**Privacidad**	**Retorno**	**Parámetros**	**Descripción**
Filtrar	Public	Dato	dato: Dato NivelDepuracion: Int	Es la interfaz de otros métodos de otras clases.
FiltrarLSBBMP	Private	Imagen	Imagen: Imagen NivelDepuración: int	Depura imágenes BMP mediante la técnica LSB.
FiltrarLSBGIF	Private	Imagen	Imagen: Imagen NivelDepuración: int	Depura imágenes GIF mediante la técnica LSB.
FiltrarLSBJPEG	Private	Imagen	Imagen: Imagen NivelDepuración: int	Depura imágenes JPEG mediante la técnica LSB.
FiltrarPaletaGIF	Private	Imagen	Imagen: Imagen NivelDepuración: int	Depura imágenes GIF mediante la técnica Gifshuffle.
Unsigned	Private	Int	b: Byte	Pasa un entero con signo a sin signo.
Signed	Private	INT	i: Int	Pasa un entero sin signo a con signo.
creaBufferedImage	Private	Buffered Image	Imagen: Image imageType: Int	Crea un BufferedImage a partir de una Image.

Tabla 93 - Métodos clase FiltroImagen

- **FiltroAudio:**

La clase FiltroAudio es utilizada para filtrar audios que tengan embebido otro mensaje mediante técnicas esteganográficas. Pertenece al paquete FiltrosEsteganográficos.

Atributos			
Nombre	**Privacidad**	**Tipo**	**Descripción**
Logger	Private	Logger	Se utiliza para mostrar mensajes de depuración por pantalla.

Tabla 94 - Atributos clase FiltroAudio

Métodos				
Nombre	**Privacidad**	**Retorno**	**Parámetros**	**Descripción**
Filtrar	Public	Dato	dato: Dato NivelDepuracion: Int	Es la interfaz de otros métodos de otras clases.
FiltrarMP3Stego	Private	Audio	audio: Audio NivelDepuración: int	Depura audios introduciendo información mediante el programa esteganográfico MP3Stego.

Tabla 95 - Métodos clase FiltroAudio

3.4 Modelado de la Arquitectura Dinámica

La arquitectura dinámica del sistema permite mostrar el funcionamiento del conjunto del sistema, mostrando sus funcionalidades, y ayudando a comprender mejor los procesos que esta realiza.

3.4.1Diagrama de Casos de Uso

El diagrama de casos de uso es un diagrama de comportamiento, en el que se define como debe comportarse el sistema. En el caso de este proyecto, la funcionalidad de la aplicación no se activa mediante la intervención de un usuario, sino mediante el inicio de una comunicación HTTP por parte de una máquina.

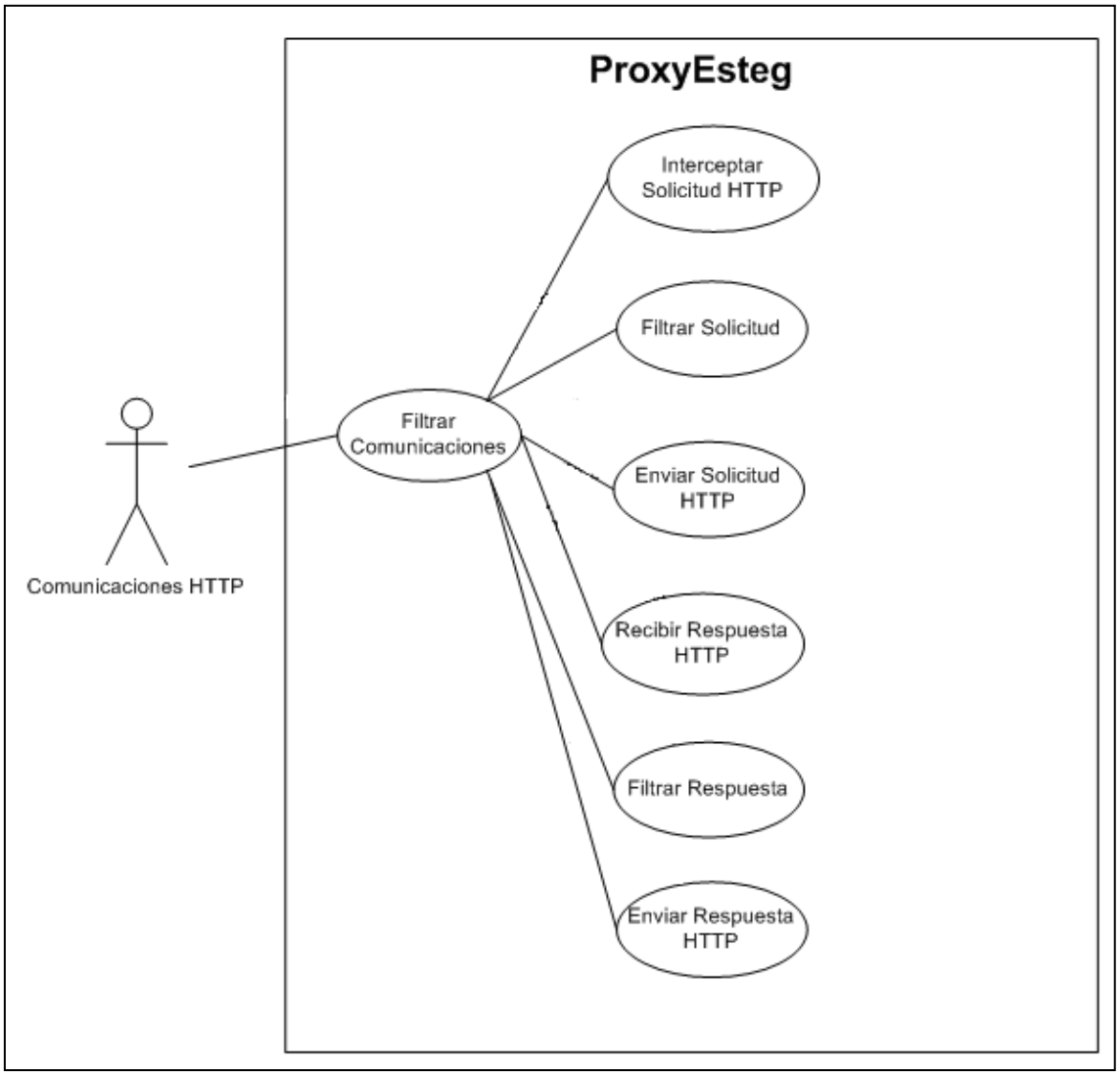

Ilustración 20 - Diagrama de casos de uso

3.4.2 Diagrama de Estados

El diagrama de estados es utilizado para mostrar el camino que un flujo de información al ejecutarse en cada proceso. A continuación se va a mostrar el diagrama de estados de la clase ConexionHTTP, por ser el núcleo del proxy HTTP, realizando la parte más importante y compleja de las acciones de la aplicación.

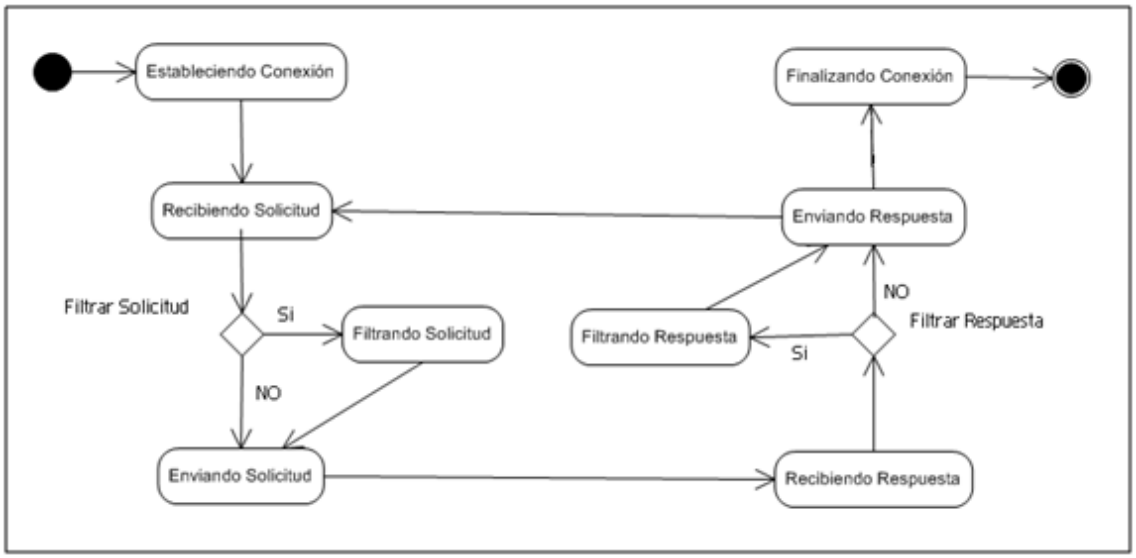

Ilustración 21 - Diagrama de Estados (Clase ConexionHTTP)

3.4.3 Diagrama de Secuencia

El diagrama de secuencia sirve para mostrar la interacción entre los objetos de la aplicación. A continuación se muestran los diagramas de los distintos casos de uso de la aplicación:

- Interceptar solicitud HTTP:

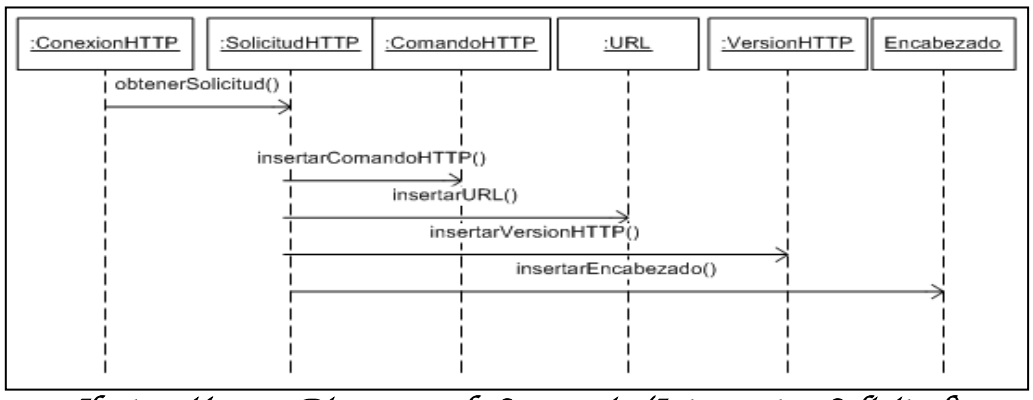

Ilustración 22 - Diagrama de Secuencia (Interceptar Solicitud)

- Filtrar Solicitud:

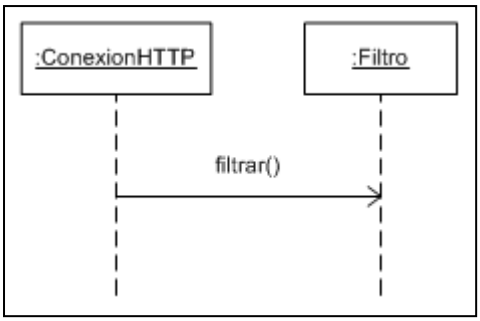

Ilustración 23 - Diagrama de Secuencia (Filtrar Solicitud)

- Enviar Solicitud HTTP:

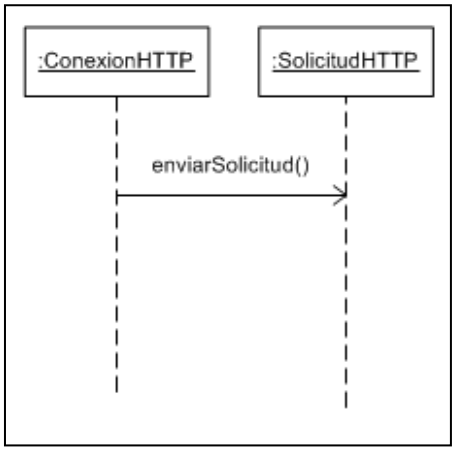

Ilustración 24 - Diagrama de Secuencia (Enviar Solicitud)

- Interceptar Respuesta HTTP:

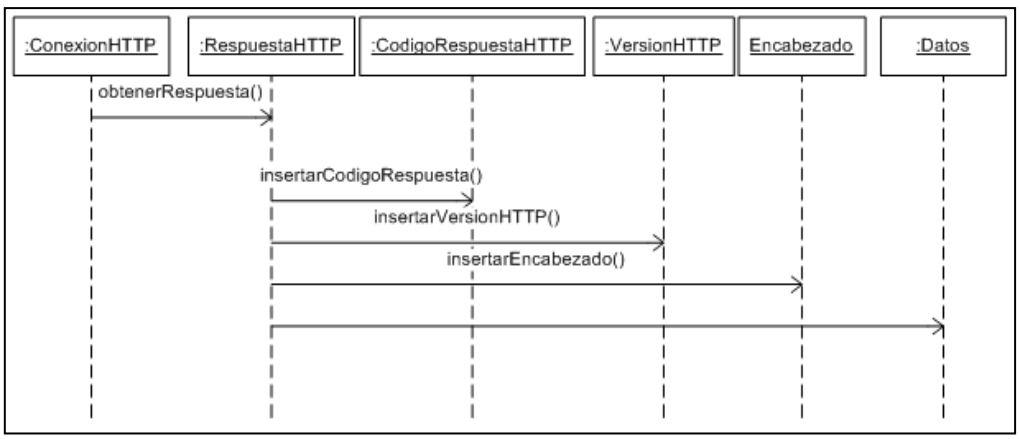

Ilustración 25 - Diagrama de Secuencia (Interceptar Respuesta)

- Filtrar Respuesta:

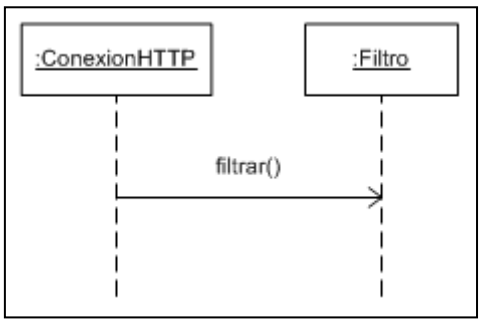

Ilustración 26 - Diagrama de Secuencia (Filtrar Respuesta)

- Enviar Respuesta:

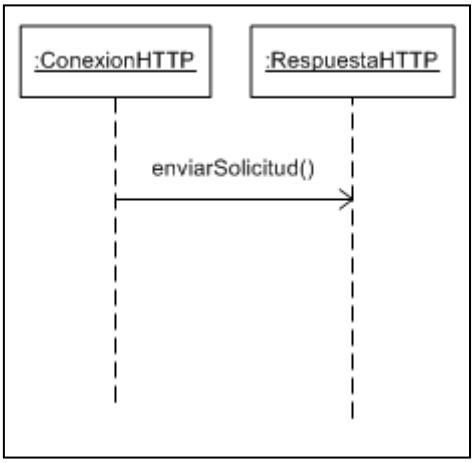

Ilustración 27 - Diagrama de Secuencia (Enviar Respuesta)

Capítulo 4
Implementación del Proyecto

En este capítulo se muestran los detalles de la implementación del proyecto, por lo tanto, se explicará tanto los detalles de la implementación del proxy HTTP, como los detalles de la implementación de los filtros esteganográficos.

Por otra parte, también se incluyen las pruebas realizadas a la aplicación una vez finalizada. Dichas pruebas se dividen en dos tipos, las pruebas de validación, definidas en la etapa de análisis y que sirven para comprobar que se cumplen con los requisitos del sistema, y las pruebas de rendimiento, utilizadas para comprobar el rendimiento de la aplicación y así saber qué diferencias de tiempos existen entre el acceso a recursos Web mediante el empleo del proxy y sin este.

4.1. Lenguaje de Programación

El lenguaje de programación con el que ha sido implementada la aplicación es Java (13). Java es un lenguaje de alto nivel orientado a objetos, con mucha sintaxis similar a la de los lenguajes C y C++, y que elimina las herramientas de bajo nivel de estos lenguajes, ya que son las que generalmente suelen inducir a errores en su manipulación, como es el caso de los punteros.

Las aplicaciones Java son compiladas normalmente a un código intermedio (llamado Java Bytecode) en vez de directamente a código máquina. Este código intermedio es ejecutado en la máquina virtual de Java, un programa escrito en código nativo de la plataforma de destino, que se encarga de interpretar el código y ejecutarlo. Esto permite que las aplicaciones sean portables a cualquier plataforma, aunque siempre puede ocurrir pequeños problemas que deben ser depurados antes de ejecutar el programa en otra plataforma distinta a la plataforma para la que fue diseñado la aplicación. Si se desea obtener un mayor rendimiento, se puede compilar la aplicación a código nativo de una plataforma, se consigue un gran aumento del rendimiento pero se pierde la portabilidad de la aplicación.

La elección del lenguaje Java para la implementación de este proyecto se ha debido a varios factores principales. Primero, por permitir trabajar con el paradigma de programación de la orientación a objetos, facilitando en gran medida el desarrollo de la aplicación, permitiendo tratar los datos transferidos mediante el protocolo HTTP como objetos, ofreciendo un manejo sencillo de los mismos. Segundo, porque el empleo de hilos se realiza de una manera muy sencilla, permitiendo que la aplicación sea escalable, mejorando el rendimiento de la misma con el aumento del número de procesadores del computador donde se ejecute. Tercero, por permitir ampliar las funcionalidades de la misma de una manera sencilla, permitiendo en un futuro la ampliación de los filtros esteganográficos, o depurando otros protocolos. Por último, la portabilidad que ofrece la máquina virtual de Java nos permite ejecutar la aplicación sobre cualquier plataforma, no obligando a crear una nueva aplicación en el caso de que se desee transportar esta a otra plataforma.

4.2. Implementación del Proxy HTTP

La implementación del proxy HTTP, parte fundamental de la aplicación, ha sido realizada mediante el empleo de hilos, usando la clase de Java "Threads". Los hilos han sido usados tanto para el código principal del proxy, encargado de interceptar las solicitudes HTTP, como para la gestión de dichas intercepciones. Primero se crea el hilo del proxy principal, que cada vez que recibe una transmisión HTTP se encarga de crear un nuevo hilo de ejecución y de decirle sobre que socket debe actuar. Estos hilos creados por el hilo principal atienden todo el proceso de la comunicación HTTP, tanto las intercepciones y envíos de las solicitudes y respuestas HTTP, como la utilización de los filtros para eliminar información oculta en estas.

Aunque se podría haber atendido cada comunicación HTTP en el hilo principal, ahorrándose el tiempo de creación de los hilos, se optó por el uso de hilos también para cada conexión HTTP con el fin de permitir atender varias conexiones simultáneas, incluidas las conexiones persistentes. Además, esto permite que la aplicación sea escalable, ya que la ejecución de la misma en un computador con varios procesadores puede permitir, que al aumentarse el número de estos, se obtenga un aumento del rendimiento de la aplicación. También se debe tener en cuenta que mediante el empleo de hilos se consigue que los tiempos que se emplean en filtrar la información enviada mediante HTTP no influye en los otros hilos, ya que son independientes unos de otros.

Para interceptar las comunicaciones se han empleado los socket de la clase "Socket" y "ServerSocket" de Java. Dichos socket permiten abrir comunicaciones entre aplicaciones, permitiendo crear la arquitectura cliente-servidor del proxy. El socket de "ServerSocket" permite esperar una petición que venga por la red, esperando dicha petición por el puerto deseado. Sin embargo, los socket de la clase "Socket" son empleados para comunicarse con el servidor Web donde se encuentran los recursos deseados, siempre conectándose al puerto número 80, puerto usado por el protocolo HTTP. Todas las transmisiones, tanto con clientes como con servidores, son realizadas a través de estos socket.

Para parametrizar el proxy se ha optado por un fichero de configuración externo, llamado "Configure", que es leído al arrancar el proxy. Los datos de los parámetros leídos en este fichero son utilizados tanto para configurar el proxy, como para configurar los filtros esteganográficos, tanto la activación como los niveles de actuación de los mismos.

4.3. Implementación de los Filtros Esteganográficos

Para la implementación de los filtros esteganográficos se ha creado una interfaz común para todos ellos, con un único método. Este método común a todos los filtros, es el encargado de, en función del tipo de dato que se desea filtrar, elegir el filtro adecuado para ese dato y usarlo para eliminar información oculta en él.

Cada filtro ha sido implementado utilizando técnicas de esteganálisis mediante el empleo de un guardián activo para eliminar información ocultada en archivos mediante las herramientas esteganográficas descritas en apartado 2.1.2 del capítulo de análisis de este documento.

4.4. Pruebas

En esta sección se detallan las pruebas realizadas a la aplicación, tanto para validar los requisitos de usuario, como la pruebas de rendimiento realizadas a la aplicación para comprobar el funcionamiento de esta. Las pruebas han sido realizadas en un computador de las siguientes características:

- Procesador Intel® Core2 Quad 2.40 GHz.
- 4094 MB de memoria RAM.
- Tarjeta de red Gigabit Ethernet.
- Conexión a Internet de 3Mb (velocidad de subida 269 Kbps, velocidad de bajada 2595 Kbps).

4.4.1. Pruebas de Validación

En este apartado está dedicado a la verificación de los requisitos de usuario. Para tal propósito, se han realizado las pruebas de validación definidas en el apartado 2.3.1 del capítulo de análisis del proyecto. En este apartado solo se comentarán los resultados de dichas pruebas. Si se desea ver los resultados concretos de cada una de las pruebas, estas se encuentran en el **"Anexo B"** de este documento.

Tras la realización de las pruebas de validación de la aplicación, y habiendo siendo sido superadas la totalidad de estas, se puede afirmar que la aplicación cumple con todos los requisitos de usuario especificados durante la fase de análisis del proyecto.

4.4.2. Pruebas de Rendimiento

En este apartado se analizan los resultados obtenidos en las pruebas de rendimiento. Los las pruebas de rendimiento están detalladas en el apartado 2.3.3 del capítulo de análisis, y los resultados de estas se encuentran en el **"Anexo C".**

Los resultados de las pruebas demuestran que existe una gran diferencia en los tiempos, tanto de acceso a recursos Web como en el envió de datos, cuando se usa el proxy y cuando no se usa el proxy, siendo necesario el doble de tiempo en el mejor de los casos cuando se usa el proxy. Ésta diferencia de tiempos se origina por el tratamiento de los datos que hace el proxy, recibiéndolos, almacenándolos y posteriormente enviándolos. Al contrario de lo que sucede en las comunicaciones cliente-servidor del protocolo HTTP, donde el cliente y el servidor se comunican directamente entre ellos, cuando se usa el proxy se producen el doble de comunicaciones, debido a que toda la información pasa a través de éste. El aumento de tiempos al realizar las solicitudes y recibir las respuestas puede considerarse normal debido al tratamiento de datos antes comentado, aunque podría reducirse éste mejorando el código de la aplicación para obtener un mayor rendimiento.

También se han realizado prueba para comprobar la diferencia de tiempos que se producen cuando se ejecuta el proxy sin filtros o con alguno de ellos. Los resultados obtenidos muestran que las diferencias de tiempos cuando se usa solo el proxy, y cuando éste tiene activado alguno de los filtros, es pequeña. Esto viene a indicar que la mayor parte del tiempo está dedicada a la recepción y envío de las solicitudes y respuestas HTTP, siendo el tiempo dedicado a filtrar la información inferior en comparación con éste otro tiempo. Otro prueba a favor del argumento de que se utiliza más tiempo para el envío y recepción de la información que para el tratamiento de la misma son las similitudes de tiempos que se producen cuando se ejecuta el proxy con el filtro de solicitudes, con el filtro de respuestas, o con ambos, siendo en todos los casos tiempos muy parecidos, diferenciados por solo unos pocos segundos, diferencia que puede ocurrir por el peso de los archivos que cada filtro tiene que filtrar, debiéndose dedicar más tiempo al filtrado del archivo cuanto más pesado es el mismo.

Para finalizar, comentar que, aunque la diferencia entre los pesos de las imágenes afectan a los tiempos del proxy. Dicho peso afecta en gran manera a los tiempos de retransmisión, aunque en el caso de los filtros esteganográficos, el peso de los archivos que estos filtran solo afectan al tiempo en una cantidad ínfimamente inferior que el tiempo empleado para su retransmisión.

Capítulo 5
Gestión del Proyecto

El capítulo "Gestión del Proyecto" ofrece, de manera detallada, la gestión de los recursos materiales, humanos y el tiempo empleados en el proyecto. Dicha gestión está desglosada en la planificación empleada para la realización del proyecto, compuesta, entre otras cosas, por un diagrama Gantt, y el presupuesto empleado durante el proyecto, detallando los bienes materiales y los recursos humanos empleados.

Esta planificación sirve para obtener información acerca del proyecto, saber la duración que ha tenido el mismo, así como los costos de los recursos empleados.

5.1 Planificación del Proyecto

En este apartado se muestra la planificación del proyecto, para ello, se ha realizado un diagrama Gantt con el fin de facilitar la comprensión de ésta. Además, acompañando al diagrama Gantt, se ofrece una descripción de las distintas fases de las que ha estado compuesto el desarrollo de este proyecto.

- **Fase de Análisis:** En esta fase se ha realizado el estudio previo del proyecto, con el fin de obtener una información en detalle de todos relacionados con el mismo. También se dedicó tiempo al entrenamiento del personal encargado del desarrollo del proyecto en temas relacionados con éste, tales como la esteganografía, los proxys HTTP, o el lenguaje de programación Java. Además, en esta fase se han desarrollado los requisitos de usuario de la aplicación, así como el plan de pruebas encargado de validar el sistema y de comprobar el correcto rendimiento del mismo.

- **Fase de Diseño:** En la fase de diseño se ha definido la arquitectura del sistema. También se ha realizado el diseño detallado de la aplicación, incluyendo tanto la arquitectura dinámica como la arquitectura estática.

- **Fase de Codificación:** Una vez realizado un diseño para la aplicación que ofreciese una solución factible y sencilla a los requisitos de usuario obtenidos en la fase de análisis, se procedió a realizar la codificación de la aplicación. La codificación de la aplicación ha sido realizada en el lenguaje de programación Java.

- **Fase de Integración:** Tras la fase de codificación, cuando la aplicación ya ha sido realizada, se pasa a la fase de Integración. En esta fase se realizan las pruebas definidas en la fase de análisis, que tienen como objetivo la validación de la aplicación así como la comprobación del rendimiento de ésta. Posteriormente, se finaliza la documentación del proyecto, ya que está habrá sido hecha poco a poco en las distintas etapas del proyecto. Una vez realizado lo anterior, ya solo falta instalarle al cliente la aplicación e impartir los cursos de formación sobre el manejo de ésta.

A continuación se muestra el diagrama Gantt con el orden y duración de las distintas fases del proyecto:

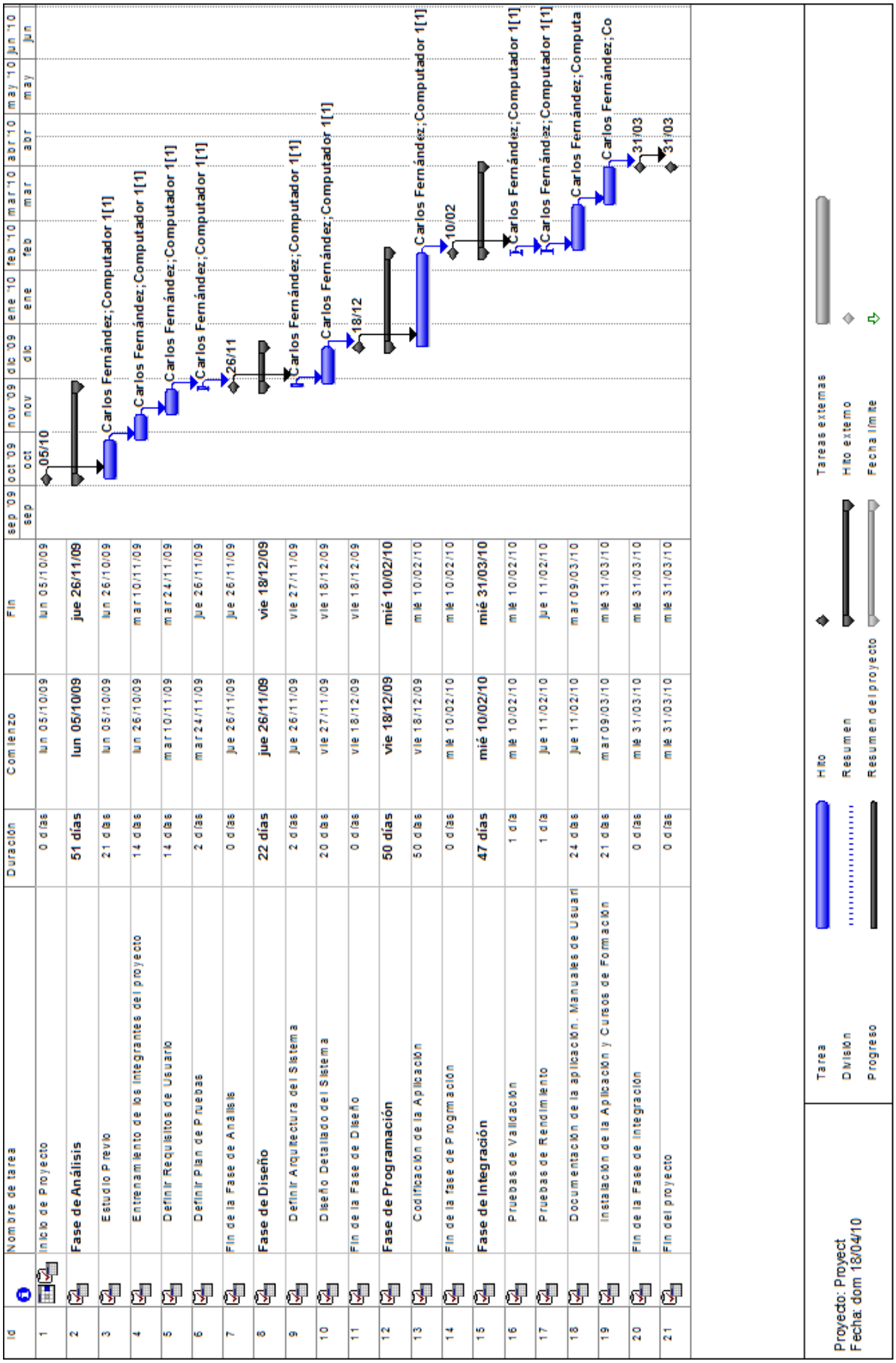

Ilustración 28 - Diagrama Gantt

5.2 Presupuesto del Proyecto

En este apartado se detalla el presupuesto empleado para la realización del proyecto. El presupuesto del proyecto está compuesto por el coste de los recursos empleados en el mismo, tanto materiales como humanos. A estos costes se le añadirá un diez por ciento de los mismos, siendo los costes indirectos del proyecto, tales como uso de electricidad, conexión ADSL, etc. Por último, al coste de la aplicación se le añadirá el impuesto sobre el valor añadido (IVA) de un 16%.

Para facilitar su comprensión, éste estará desglosado, diferenciando el coste de los recursos humanos y el de los recursos materiales, sumándole a todo esto los costes indirectos del proyecto. También se presentará el coste total sin IVA y con IVA.

El coste del uso computador utilizado para el desarrollo del proyecto es de 166,67€, siendo el coste total de este 1.000€, el tiempo de amortización 3 años y se ha usado el mismo durante 6 meses. El coste de la persona encargada del desarrollo completo del proyecto es de 30 €/hora, habiendo trabajado ésta un total de 1020 horas. Los costes indirectos han sido considerados el 10% del presupuesto, y el IVA aplicado es del 16%.

Se ha optado por no incluir en el presupuesto ningún beneficio por el desarrollo del mismo, debido ha sido realizado para su venta. En el caso de que exista un interés por la compra de éste, se deberán añadir los beneficios al presupuesto previa aplicación del impuesto sobre el valor añadido (IVA).

1020 horas x 30 €/hora = 30.600,00 €

Tabla 96 - Calculo Coste Recursos Humanos

Concepto	Coste (Euros)
Recursos Humanos	30.600,00 €
Recurso Materiales	166,67€
Costes Totales	30.766,67 €
Costes Indirectos (10%)	3.076,67 €
Presupuesto Sin IVA	**33.843,34 €**
IVA (16%)	5.414,93 €
Presupuesto Final	**44.673,20 €**

Tabla 97 - Presupuesto del Proyecto

Capítulo 6
Conclusiones y Trabajos Futuros

Este capítulo sirve para exponer las conclusiones obtenidas durante la realización del proyecto, sirviendo de resumen final de este. Así mismo, se narran los posibles trabajos futuros que se pueden realizar sobre el proyecto, pudiendo ser meros perfeccionamientos de la aplicación, a posibles ampliaciones en la funcionalidad o expansiones del mismo a otras áreas relacionadas con las del proyecto.

6.1 Conclusiones

Tras la finalización del proyecto, y la realización de las pruebas creadas para la validación del mismo, se puede afirmar que se ha cumplido con todos los objetivos propuestos inicialmente. La aplicación es capaz de interceptar las comunicaciones HTTP entre clientes y servidores, permitiendo así el filtrado de los datos transmitidos para evitar el envío de información o archivos, dentro de otros archivos, mediante el uso de técnicas esteganográficas. De esta manera, la aplicación puede servir como punto de seguridad de la red, filtrando los datos de las comunicaciones que entren y salgan de la misma.

Además, se ha conseguido que la aplicación tenga un buen rendimiento, haciendo que la intercepción de las comunicaciones por parte del proxy no aumente en demasía el tiempo de acceso a recursos Web. Por otra parte, se produce una pequeña caída de rendimiento cuando se envían datos pesados, y sobre todo cuando estos han de pasar por los filtros esteganográficos. Esto se produce debido a que los algoritmos de dichos filtros deben modificar en algunos muchos bytes de la información transmitida, provocando la caída de rendimiento, tardándose más de lo habitual en realizar la tarea.

En cuanto el diseño de la aplicación, se ha logrado que este sea lo más reutilizable posible. Esto puede permitir la reutilización del proxy HTTP y de los filtros esteganográficos, permitiendo que el proxy pueda ser reutilizado para otros proyectos en los que se necesite analizar las comunicaciones HTTP entre clientes y servidores Web. Además, los filtros esteganográficos están diseñados para poder ser empleados en otras aplicaciones sin la necesidad de efectuar cambios en los mismos. Esto se ha logrado haciéndolos totalmente independientes del proxy HTTP, de manera que el paquete de clases de los filtros puede ser exportado a otros proyectos para usar los métodos que contiene.

La implementación de este proyecto mediante el empleo del lenguaje de programación Java, ha permitido a la aplicación ser independiente de la plataforma (son necesarios algunos arreglos para su correcto funcionamiento en algunas plataformas distintas a la plataforma para la que fue diseñada la aplicación, Windows ®, no se garantiza el correcto funcionamiento de los filtros esteganográficos de audio fuera de dicha plataforma), además de facilitar la reutilización de la aplicación mediante el empleo de paquetes para agrupar las distintas clases que conforma la aplicación.

También cabe destacar el buen rendimiento de la aplicación, que aunque en algunos casos se producen grandes retardos en el tiempo de retransmisión de la información, ofrece un rendimiento bueno en el caso del acceso normal a páginas Web, produciéndose los mayores retardos en los tiempos cuando se

envían datos, como en el caso del comando POST de HTTP, siendo mayores los retardos cuanto mayor es la información enviada.

Para finalizar, comentar que la aplicación ha sido diseñada para facilitar el empleo de la misma al usuario, por lo que la configuración y el arranque de esta se realiza de una manera fácil, sencilla e intuitiva.

- Conclusiones Personales:

Elegí realizar este proyecto tras proponérmelo mi tutor, habiendo ido a hablar con el buscando información sobre otro proyecto, también sobre esteganografía, que había llamado mi atención, al tratarse de un tema desconocido para mí, a la par de ser un tema muy interesante y actual.

Este proyecto me ofrecía la oportunidad de aprender sobre temas muy diversos, como la esteganografía en la seguridad informática, las comunicaciones entre máquinas mediante el protocolo HTTP, así como practicar con otros conocimientos adquiridos durante estos años, como el uso puede ser el empleo de hilos para ejecutar paralelamente un programa, así como las comunicaciones mediante el uso de sockets.

Para finalizar, resaltar otra de las razones que he tenido para elegir realizar este proyecto fue aprender a programar en Java, lenguaje de programación orientado a objetos, que no había tenido la oportunidad de aprender durante estos años de carrera, y que es muy empleado en el mundo empresarial.

6.2 Trabajo Futuro

Aunque se han cumplido con todos los objetivos, y se ha creado la aplicación siguiendo los requisitos establecidos durante la fase de análisis del proyecto, esta puede ser mejorada y ampliada para ofrecer nuevas funcionalidades. En este apartado se va a estudiar los posibles nuevos objetivos que se pueden plantear con el fin de mejorar la aplicación, ya sea para mejorar el rendimiento de esta, como para ampliar su funcionalidad. A continuación se ofrecen los distintos objetivos que un trabajo futuro sobre la aplicación podría llevar a cabo:

- Mejorar la implementación de los algoritmos que conforman la aplicación, tanto los del subsistema del proxy como del subsistema de los filtros esteganográficos, con el fin de mejorar el rendimiento de la aplicación y reducir el tiempo dedicado al envío y recepción de las comunicaciones.

- Ampliar el número de filtros esteganográficos disponibles. La creación de estos nuevos filtros deberá ofrecer una ampliación de los tipos de datos para los que se ofrece protección, así y como una mayor variedad de filtros para distintos tipos de técnicas esteganográficas. Se deberá ampliar los filtros para archivos de audio e imágenes, y crear nuevos filtros para video y texto. Además se podrá diseñar filtros para el protocolo HTTP, es decir, filtros para eliminar información oculta en los mismos datos que emplea el protocolo.

- Ampliar el proxy esteganográfico mediante la creación de otros proxys que permitan interceptar las comunicaciones de red realizadas por otros protocolos, con objeto de filtrar los datos enviados usando los filtros esteganográficos existentes. Otros protocolos susceptibles de ser implementados para ampliar el servicio que ofrece el proxy esteganográfico son, por ejemplo, FTP y SMTP, siendo estos protocolos muy utilizados para la transmisión de datos, aunque bien se podría crear para cualquier otro protocolo.

- Mejorar la parametrización de la aplicación, permitiendo al usuario elegir más opciones para permitir una configuración más personalizada al usuario. Por ejemplo, se podría permitir configurar los filtros de forma individual, indicando en cada caso si dicho filtro está activo y el nivel de actuación del mismo.

- Mejorar la aplicación con el objetivo de mostrar los errores de ejecución de forma clara y precisa. Mejorando esta funcionalidad, se conseguiría facilitar a los desarrolladores la depuración de los fallos que se puedan producir durante la ejecución de la aplicación.

En el "Anexo D" se ofrecen ideas de cómo ampliar el número de filtros esteganográficos de la aplicación, permitiendo que si en un futuro se desea ampliar el número de éstos, se tenga una pequeña guía para lograrlo de la forma más sencilla posible.

Bibliografía

1. **Kutter, M. y Petitcolas, F. A. P.** Peticolas. *Peticolas.* [En línea] http://www.petitcolas.net/fabien/publications/ei99-benchmark.pdf.
2. **Cox, Ingemar, et al.** *Digital Watermarking and Steganography, 2nd Ed.* s.l. : Morgan Kaufmann. 978-0123725851.
3. **Kwan, Matthew.** Darkside Technologies. *Darkside Technologies.* [Online] http://www.darkside.com.au/gifshuffle/index.html.
4. **Gibson, Tyler.** Snotmonkey. [En línea] Marzo de 1010. http://www.snotmonkey.com/.
5. **Collomosse, John.** Blindside. [Online] Enero 2010. http://www.blindside.co.uk.
6. **Latham, Allan.** JPHS. [En línea] http://linux01.gwdg.de/~alatham/stego.html.
7. **Petitcolas, Fabien.** Peticolas. *Peticolas.* [En línea] http://www.petitcolas.net/.
8. **The Internet Society.** [En línea] 1999. http://www.ietf.org/rfc/rfc2616.txt.
9. Wikipedia. [En línea] http://es.wikipedia.org/wiki/Wikipedia:Portada.
10. **Hughes, Merlin, Shoffner, Michael y Hamner, Derek.** *Java Network Programing.* s.l. : Ed. Manning. 188477749X.
11. **Giner, José Manuel.** Proxy Anonimo. [En línea] http://proxyanonimo.es/.
12. **Sarc.** Sarc. [Online] Back Bone Security, 2004. http://www.sarc-wv.com/.
13. **Sun Microsystems.** [En línea] Marzo de 2010. http://es.sun.com/.

Anexo A: Manual de la Aplicación

En este anexo se detalla el manejo de la aplicación, dividiendo dicho manejo en cinco partes fundamentales. Los elementos adjuntos al documento del proyecto. La instalación de la aplicación, donde se explica que debe hacer el usuario para poder emplear de la aplicación. Los requisitos mínimos, donde se especifica las características mínimas que debe tener la máquina donde se quiera ejecutar la aplicación, así como otras aplicaciones necesarias. Después se detalla la configuración y puesta en marcha de la aplicación, donde se enseña a configurar esta, dependiendo de las tareas que se desea que realice la aplicación, antes de ponerla en funcionamiento. Por último, se enseñarán los distintos errores que puede mostrar la aplicación durante el transcurso de su uso. Dichos errores podrán ser empleados para realizar mejoras en futuras versiones de la aplicación.

A continuación, se procede a detallar las distintas partes del manual antes detalladas:

- **Elementos adjuntos:**

Junto a este documento debe venir adjunto un CD con los archivos del proyecto, entre ellos:

- ✓ El archivo **"ProxyEsteg.rar",** que contiene el ejecutable .jar, además de todo lo necesario para el correcto funcionamiento de la aplicación. Su instalación viene detallada más adelante.
- ✓ El archivo "Memoria – ProxyEsteg.pdf", que es una copia en formato electrónico de este mismo documento.
- ✓ El archivo "Codigo – ProxyEsteg.rar", que contiene el código implementado para la creación de esta aplicación.
- ✓ Además, se incluye también un archivo comprimido de cada uno de los programas esteganográficos usados para la elaboración de los filtros del proyecto, a saber: JPHIDE, BSIDE, GIFSHUFFLE y MP3Stego. Todos ellos han sido creados para la plataforma Windows ©.

- **Instalación de la Aplicación:**

Para la instalación de la aplicación se debe copiar al disco duro del computador donde se desea instalar ésta el archivo ProxyEsteg.rar. Dicho archivo se encuentra en el CD que se adjunta junto a este documento.

La instalación de la aplicación es sumamente sencilla. Una vez copiado el archivo al disco duro, solo es necesario extraer la carpeta que éste contiene en la ruta donde se desea instalar la aplicación.

- **Requisitos Mínimos:**

En esta sección muestran los requisitos mínimos de la aplicación:

- Windows XP ®, Windows Vista ®.
- Máquina Virtual de Java.
- Tarjeta Ethernet, router y conexión a Internet.

Nota: Para que el programa pueda ser correctamente ejecutado, es necesario la aplicación "ffmpeg" y la aplicación "MP3Stego". El filtro para mp3 no funcionará sin ambas aplicaciones, ya sea por su ausencia, o porque estás no puedan ser ejecutadas fuera de la plataforma para la que están diseñadas, Windows ®.

- **Configuración y Puesta en Marcha:**

La configuración de la aplicación se realiza a través del fichero "Configure". Dicho fichero se encuentra situado en el directorio de instalación de la aplicación. El fichero está compuesto por seis líneas, cada una dedicada compuesta por una cabecera que indica el campo al que se desea dar valor, seguida de dos puntos y el valor que se desea separado por un espacio. A continuación se muestra en una tabla, para cada cabecera de configuración posible, una descripción detallada de la misma y los valores que esta puede tomar.

Cabecera	Descripción	Valores Posibles
Puerto:	Indica el número del puerto por el que se realizan las peticiones HTTP.	Puerto>0
Depuración solicitud activa:	Tendrá el valor "SI" si se desea que la depuración de solicitudes esté activa y "NO" en caso contrario.	["SI","NO"]
Depuración respuesta activa:	Tendrá el valor "SI" si se desea que la depuración de respuestas esté activa y "NO" en caso contrario.	["SI","NO"]
Nivel depuración solicitud:	Indica el nivel de depuración del filtro, es decir, el nivel indica los bits modificados por byte, empezando por el menos significativo en adelante.	8≥Nivel≥0
Nivel depuración respuesta:	Indica el nivel de depuración del filtro, es decir, el nivel indica los bits modificados por byte, empezando por el menos significativo en adelante.	8≥Nivel≥0
Nivel Logger	Indica el nivel que el logger de la API de java utilizará para mostrar mensajes por pantalla, desde información a errores graves.	"OFF", "INFO", "WARNING", "SEVERE", "ALL", "FINEST", "FINER", "FINE", "CONFIG"

Por defecto, el proxy se conectará al puerto 80, la depuración de las solicitudes estará activa con nivel 1 y la depuración de las respuestas estará desactivas. El nivel del logger por defecto es "OFF".

Nota: El parámetro "Nivel Logger" sirve para la depuración de los errores producidos por aplicación por parte de los diseñadores de la misma. Se recomienda a los usuarios de la aplicación que el valor de este parámetro este fijado en "OFF" o en "SEVERE" en caso de que se desee ver los errores graves de la aplicación, a fin de no verse saturados de información no relevante.

Una vez instalada la aplicación según lo expuesto anteriormente, la puesta en marcha de la aplicación se realizar mediante la ejecución en la consola de comandos del sistema operativo del siguiente comando:

```
java –jar ProxyEsteg.jar
```

- **Errores de la Aplicación:**

Los errores mostrados por la aplicación tienen labor puramente depurativa, ya que la presencia de estos no provoca el fin del proceso en ejecución de la aplicación. Sin embargo, dichos errores provocan fallos en la ejecución del proxy, pudiendo no hacer funcionar una petición HTTP a una página Web, con la consecuente imposibilidad de visualizar dicha página, o no filtrando un dato mediante el filtro esteganográfico correspondiente, no haciendo correctamente su trabajo de eliminación de información esteganográfica para un dato concreto.

Anexo B: Pruebas de Validación

En este apartado se muestran los resultados de las pruebas de validación de la aplicación. Para mostrar dichos resultados se va a mostrar una tabla por cada una de las pruebas realizadas, con una descripción concreta de los datos usados para realizar la prueba, además del resultado de la misma. Cada prueba contará con un identificador, siendo este el identificador de la prueba realizada.

PRU-001	
Descripción concreta de la prueba:	Se ha accedido a la página Web www.elpais.es y se ha esperado a que esta se cargara en el navegador Web.
Resultado	Correcto

PRU-002	
Descripción concreta de la prueba:	Mediante una cuenta de correo gratuita en www.hotmail.com, se ha enviado un archivo de texto en un correo y se ha comprobado su correcta recepción.
Resultado	Correcto

PRU-003	
Descripción concreta de la prueba:	Se ha accedido a www.elpais.es y a www.hotmail.com de manera simultánea.
Resultado	Correcto

PRU-004	
Descripción concreta de la prueba:	Se ha accedido a www.elpais.es con el filtrado de respuestas activado, comprobando el correcto cargado de la página en ambas ocasiones.
Resultado	Correcto

PRU-005	
Descripción concreta de la prueba:	Se ha accedido a www.elpais.es con el filtrado de respuestas desactivado, comprobando el correcto cargado de la página en ambas ocasiones.
Resultado	Correcto

PRU-006	
Descripción concreta de la prueba:	Mediante una cuenta de correo gratuita en www.hotmail.com, se ha enviado una imagen en un correo, con el filtrado de solicitudes activado, siendo un éxito el proceso.
Resultado	Correcto

PRU-007	
Descripción concreta de la prueba:	Mediante una cuenta de correo gratuita en www.hotmail.com, se ha enviado una imagen en un correo, con el filtrado de solicitudes desactivado, siendo un éxito el proceso.
Resultado	Correcto

PRU-008	
Descripción concreta de la prueba:	Se ha accedido a la página Web www.google.es, modificando la configuración de la aplicación probando que ofreciese servicio por los puertos 80 y 8080.
Resultado	Correcto

PRU-009	
Descripción concreta de la prueba:	Mediante una cuenta de correo gratuita en www.hotmail.com, se ha enviado una imagen en un correo, con el filtro de solicitudes activo. Esto se ha realizado con el filtro en nivel 1 y posteriormente en nivel 2.
Resultado	Correcto

PRU-010	
Descripción concreta de la prueba:	Se ha accedido a www.google.es, con y sin proxy, comprobando que las imágenes cargadas no se muestran alteradas a simple vista.
Resultado	Correcto

PRU-011	
Descripción concreta de la prueba:	Enviar una imagen GIF esteganografiada con la aplicación Gifshuffle mediante una cuenta de correo en www.hotmail.com, comprobando que tras pasar el filtro esteganográfico la información oculta ha sido eliminada.
Resultado	Correcto

PRU-012	
Descripción concreta de la prueba:	Enviar una imagen GIF esteganografiada con la técnica LSB mediante una cuenta de correo en www.hotmail.com, comprobando que tras pasar el filtro esteganográfico la información oculta ha sido eliminada.
Resultado	Correcto

PRU-013	
Descripción concreta de la prueba:	Enviar una imagen JPEG esteganografiada con la aplicación JPHIDE mediante una cuenta de correo en www.hotmail.com, comprobando que tras pasar el filtro esteganográfico la información oculta ha sido eliminada.
Resultado	Correcto

PRU-014	
Descripción concreta de la prueba:	Enviar una imagen BMP esteganografiada con la aplicación BlindSide mediante una cuenta de correo en www.hotmail.com, comprobando que tras pasar el filtro esteganográfico la información oculta ha sido eliminada.
Resultado	Correcto

PRU-015	
Descripción concreta de la prueba:	Enviar un archivo de audio MP3 esteganografiada con la aplicación MP3Stego mediante una cuenta de correo en www.hotmail.com, comprobando que tras pasar el filtro esteganográfico la información oculta ha sido eliminada.
Resultado	Correcto

PRU-016	
Descripción concreta de la prueba:	Enviar una imagen, con niveles de filtrado 1 y 2 para el filtro de solicitudes, comprobando que no existen diferencias significativas con la imagen original.
Resultado	Correcto

PRU-017	
Descripción concreta de la prueba:	Enviar un archivo de audio, con niveles de filtrado 1 y 2 para el filtro de solicitudes, comprobando que no existen diferencias significativas con el audio original.
Resultado	Correcto

PRU-018	
Descripción concreta de la prueba:	Probar los parámetros de la aplicación introduciendo: • Números negativos para el proxy. • Números negativos o mayores de 8 para los niveles de filtrado. • Parámetros incorrectos para el resto.
Resultado	Correcto

PRU-019	
Descripción concreta de la prueba:	Comprobar el correcto funcionamiento de la parametrización del fichero de configuración "Configure".
Resultado	Correcto

Anexo C: Pruebas de Rendimiento

En este anexo se muestran los resultados de las pruebas de rendimiento realizadas a la aplicación tal y como están definidas en el apartado 2.2.3 del capítulo de análisis de este proyecto. La tabla mostrada a continuación contienen los resultados (mostrados en segundos) de dichas pruebas.

	Sin proxy	Proxy	Proxy y Filtro solicitudes	Proxy y filtro respuestas	Proxy y ambos filtros
www.google.es	0.9 s	1.6 s	1.8 s	1.9 s	1.7 s
www.hotmail.com	2.6 s	3.7 s	3.9 s	3.7 s	3.6
www.elpais.es	3.5 s	6.0 s	6.4 s	8.4 s	8.2 s
Envío de imagen GIF con www.hotmail.com	9.4 s	1:19.7 s	1:26.8 s	1:28.9 s	1.24.2
Envío de imagen BMP con www.hotmail.com	2.3 s	6.2 s	4:01.2 s	3:57.2 s	3:57.7 s
Envío de imagen JPEG con www.hotmail.com	2.9 s	7.0 s	8.7 s	9.3 s	8.9
Envío de archivo de audio MP3 con www.hotmail.com	6.4 s	1:24.2 s	1:33.4 s	1:28.9 s	1:38.0

Nota: Los pesos de los archivos utilizados para la toma de tiempos son: 172KB el archivo de audio MP3, 43.6KB la imagen JPEG, 280KB la imagen BMP, 165KB la imagen GIF.

Anexo D: Ampliar la Aplicación

En este anexo se explica cómo ampliar la aplicación con nueva funcionalidades. Existen dos tipos de ampliación de las funcionalidades que se pueden llevar a cabo, ampliar la aplicación para que haga de intermediaria entre las comunicaciones de otros protocolos, o ampliar el número de filtros disponibles para eliminar información oculta en archivos. A continuación se explican los pasos a seguir para conseguir cada una de las ampliaciones:

- **Ampliar Proxy para otros protocolos:**

Para diseñar e implementar otro proxy que haga de intermediario en las comunicaciones de realizadas mediante ese protocolo primero se debe analizar minuciosamente el funcionamiento del mismo, como un primer paso para comprenderlo e implementarlo correctamente. La implementación deberá utilizar los tipos de datos mostrados en el paquete "Archivos" de este proyecto para su implementación y el correcto funcionamiento de los filtros. Se deberá crear un método en la clase del proxy que contenga los datos de la comunicación distintos métodos que permitan filtrar los datos que circulen a través de ésta. Dichos métodos invocarán el método filtrar con el fin de eliminar la información oculta en los datos manejados.

Una vez realizada la implementación del proxy, se deberá unir a éste proyecto mediante la incorporación de las líneas de código necesarias para que sea lanzado como un hilo independiente de ejecución. Actualmente, esta acción se realiza en la clase "Proxy" del paquete homónimo. De esta forma se permite la ejecución simultánea de distintos proxys, cada uno encargado de actuar sobre las comunicaciones de un protocolo.

- **Ampliar Filtros Esteganográficos:**

Para ampliar los filtros esteganográficos se debe estudiar las técnicas de esteganálisis activo que se desean aplicar a los datos, así como diseñar un algoritmo que pueda ser implementado para la aplicación. Si el filtro del tipo de datos no ha sido creado todavía, se debe crear una clase dentro del paquete "FiltrosEsteganograficos", que extienda la clase "Dato", y que tenga un método llamado "Filtrar". Este método será el encargado de invocar los distintos

algoritmos para la eliminar la información oculta en los archivos, debiéndose añadir un método para cada uno de los algoritmos que se deseen implementar.

Actualmente, los algoritmos desarrollados la eliminación de información oculta son seleccionados para su uso en función del tipo de formato concreto del archivo (por ejemplo: .bmp, .gif, .mp3), pero en el caso de haber varios algoritmos para el filtrado de un archivo de un formato concreto se pueden desarrollar diversas estrategias para tratar con éste. Estas estrategias pueden ser pasar el archivo por todos los filtros, seleccionar aleatoriamente que filtros debe o no pasar, o seleccionar los filtros mediante algún método de análisis previo del archivo.